COMPUTER AUTOMATION

816 CONTROLLER REFERENCE MANUAL

COMPUTER AUTOMATION, INC.

COMPUTER
AUTOMATION
INCORPORATED
895 WEST SIXTEENTH STREET
NEWPORT BEACH, CALIF. 92660
TELEPHONE: (714) 642-9630

DECEMBER 1968

816 CONTROLLER REFERENCE MANUAL

PRINTED IN U S A

CONTENTS

Contents - continued

816

PROGRAMMED DIGITAL CONTROLLER

816 Controller - Front View

I GENERAL DESCRIPTION

INTRODUCTION

The Model 816 Programmed Digital Controller is a digital, stored-program unit that utilizes integrated circuits and a 3D core memory for simplicity and reliability. The moderate speed of the memory improves operating margins and allows powerful instructions to be implemented with ease in the processor. Parallel organization and over 140 instructions provide general-purpose computer power and flexibility. The 816 is designed for commercial and industrial control and monitoring applications where emphasis is on reliability, flexibility and economy. Examples of the uses of the 816 include:

- Central control station for inter-city communications network for concentrating and distributing messages.
- Controller for plotting tables and optical scanners.
- Automation of production line logic element testing.
- Remote-site valve control and monitoring.
- Mass spectrometer controller.

The 816 Controller has the following characteristics:

- Parallel processing
- Seven hardware registers
- 4096-word (a word is 16 bits) memory expandable to 16,384 words
- Over 140 basic instructions
- Binary, 2's complement arithmetic
- 8-microsecond memory cycle
- Block input/output from memory standard
- Three hardwired priority interrupt lines standard
- Relative, Indirect and Indexed addressing
- Automatic memory scan standard
- Immediate instructions
- Optional features

  Real-time clock

  Power fail/restart

  Priority interrupt module

Buffered output channels

Gated input channels

Modem interfaces

- Peripheral equipment

  ASR-33 Teletype with paper tape reader and punch

  High speed paper tape reader and punch

  Fixed head disc storage unit, 16,000 to 131,000 words

  Magnetic tape deck interfaces

- Software

  Symbolic assembler, 2 and 3 pass

  Debug package

  Diagnostic package

  Math library

- All silicon semiconductors
- Operating temperature range: 0° to 45°C
- Power 250 watts, approximately
- Dimensions: 8-3/4 in. high, 19 in. wide, 17 in. deep
- Weight: 40 lbs., including power supply

816 PROCESSOR

The 816 Controller contains seven hardware registers, the adder unit and the control section. Refer to Figure 1-1.

The adder is a 16-bit parallel-add, serial-carry unit utilizing complex function TTL integrated circuits. Two 16-bit words are presented to the adder unit via the S and U buses. The sum appears on the A bus, which distributes it to the W, M, P, A and X registers. The sum is then strobed into the desired register by a set or load pulse. Control of the adder is achieved by controlling the contents of the S and U buses. The output of the adder can be shifted left or right by selection gates between the adder and the A bus.

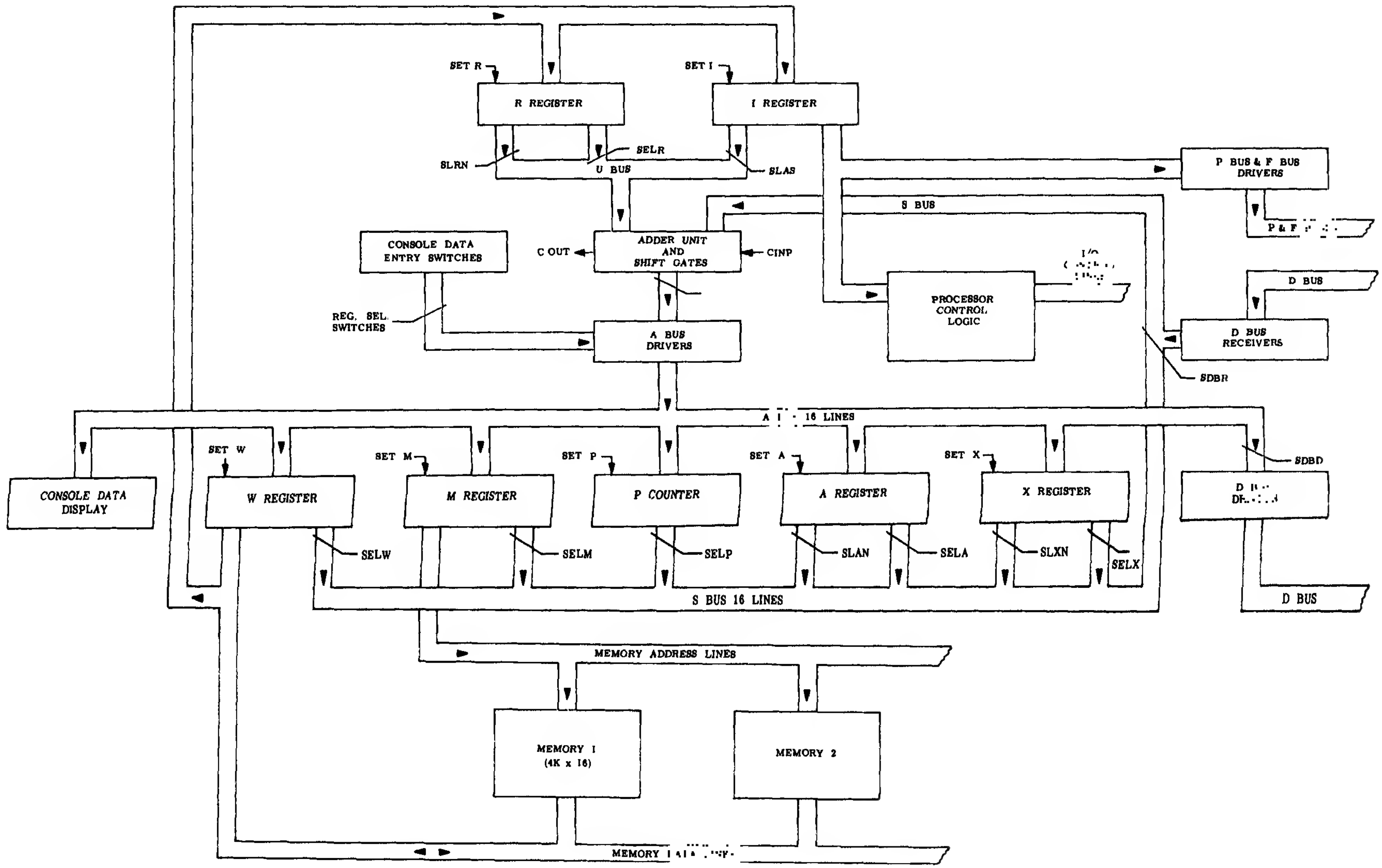
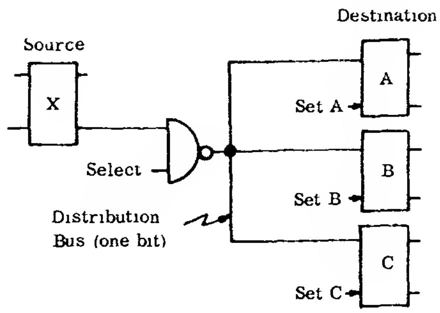


Figure 1-1 816 Controller Block Diagram

One bit of a typical bus structure is shown below:



The information on the buses is under the control of the processor control logic.

By placing the contents of one register on a bus, it may be transferred to another register by routing it through the various buses and/or adder unit until it appears as an input to one or more registers, and then strobing it into the desired register.

- The W Register is a 16-bit register that interfaces the processor to the data circuits in memory. Data read from or stored in memory is held in the W Register during the memory cycle. The W Register is always cleared at the beginning of a memory cycle. If data is to be stored in memory, it is strobed into W halfway through the cycle. If data is being read from memory, it is placed in W approximately 2 to 3 microseconds after the beginning of the memory cycle and held there during the restore portion (last half) of the cycle.

- The M Register is a 16-bit register that interfaces the processor to the address decoding circuits in the memory. Address information is stored in the M Register at the beginning of the memory cycle, where it is held throughout the cycle.

- The P Counter is a 16-bit register that serves as the program counter. It is used to hold the memory location (address) of the next instruction word in the program.

- The A Register is a 16-bit register that is used as the accumulator for arithmetic operations and serves as a word buffer register for programmed data transfers to or from I/O devices.

- The X Register is a 16-bit register that is used as an index register and also as a word buffer register for programmed transfers to or from I/O devices.

- The I Register is a 16-bit register that holds the current instruction being executed.

- The R Register is a 16-bit register that serves as an operand register to hold operands used in Memory Reference instructions.

816 MEMORY

The basic controller memory is a conventional 4-wire, 3-D core memory with an 8-microsecond full cycle time. Access time is around 2 microseconds. The basic configuration is 4096 words of 16 bits, and expansion is in 4096-word blocks up to 16K words.

A power fail-safe option is available to prevent loss of memory as power collapses due to a primary AC power failure or a power turn-off. If power is detected, an interrupt occurs which allows entry into a subroutine to effect an orderly halt to operations.

MEMORY ADDRESSING

The memory is random access requiring 14 bits of address (for 16K). The address is supplied by the M Register in the processor.

There are several modes of memory addressing to obtain the 14-bit effective address. The modes are specified by the address modifier bits (bits 8 through 10) of the memory reference instruction. See Figures 1-3 and 1-4.

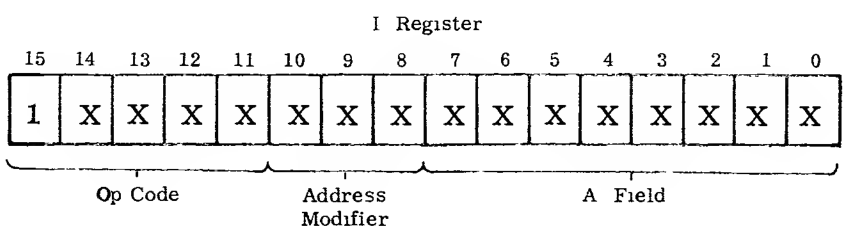


Figure 1-3 Memory Reference Instruction Format

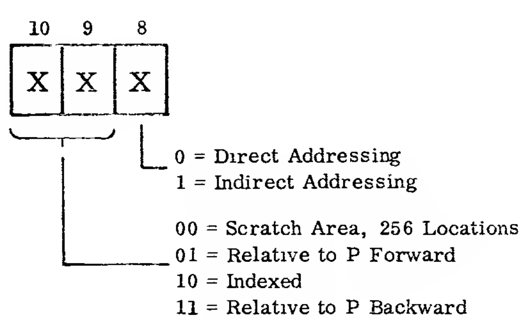


Figure 1-4 Address Modifiers

Direct Addressing

00 Scratchpad. The A field is used as the effective address in the scratch area (first 256 locations in memory)

01 Relative to P forward. The A Field is added to the current contents of the P counter and the sum is used as the effective memory location.

10 Indexed. The A Field is added to the contents of the X Register and the sum is used as the effective memory location.

11 Relative to P backward. The A Field is subtracted from the current contents of the P counter and the difference is used as the effective memory locations.

Indirect Addressing

The above methods of direct addressing allow an operand (or an instruction in the case of Jump) to be addressed directly by the instruction. In some cases, however, the location of the operand is specified by an indirect pointer - a word in memory which contains the address of the operand. This is particularly useful if the location of the desired operand is subject to change. The instruction can specify the location of the pointer and the pointer then specfies the location of the operand. This is indirect addressing. There can be several levels of indirect addressing, i.e., "multi-level indirect addressing." If the most significant bit of the indirect pointer is a zero, the pointer is the address of the operand. If the most significant bit of the pointer is a one, the pointer is the address of another pointer.

00 Scratchpad. The A Field is used as the effective address of the pointer in the scratch area (first 256 locations in memory.) The pointer is used as the address of the operand.

01 Relative to P forward. The A Field is added to the current contents of the P counter and the sum is used as the effective address of the pointer. The pointer is then used as the address of the operand.

10 Indexed. The A Field is used as the effective address of the pointer in the scratch area. The contents of the X Register is added to the contents of the pointer and the sum is used as the effective address of the operand. The contents of the pointer in memory is unchanged and X Register is unchanged.

11 Relative to P backward. The A Field is subtracted from the current contents of the P counter and the difference is used as the address of the pointer.

Indirect addressing adds one memory cycle for each level. Indexing or going relative to P does not add to execution time.

BLOCK I/O ADDRESSING

The Block input/output instructions also involve addressing memory, but always indirectly. That is, the instruction word does not contain addressing information. Instead each Block instruction has a pair of memory words associated with it. One location is used as a word counter and the other is used as an address counter. The word counter is incremented each transfer and tested for carry to indicate end-of-block. The address word is incremented each transfer and the incremented value used as the operand address. Thus Block I/O are addressing instructions, but of a special type.

II PROCESSOR INSTRUCTIONS

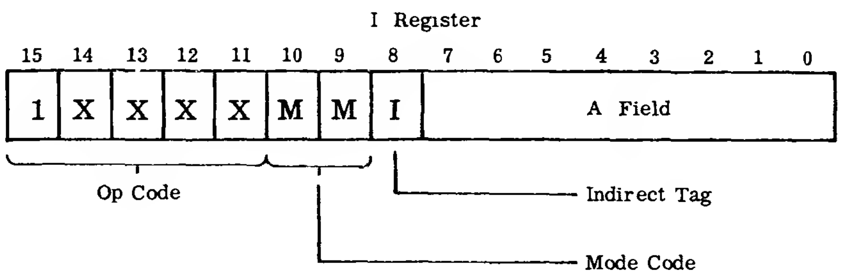
INTRODUCTION

This section describes the 816 Controller basic instruction set except for the I/O instructions which are described in the next section.

There are six classes or groupings of instructions: Memory Reference, Immediate, I/O, Conditional Jumps, Register Change and Control. All instructions are single word, and most require only one memory cycle to excute.

MEMORY REFERENCE

The format of the Memory Reference instructions is shown below. The first eight bits of the word contains the Operation Code and the address modifiers. The last eight bits of the word contain the A-Field which is used to specify or augment a memory address.



MEMORY REFERENCE FORMAT

The indirect tag specifies direct or indirect addressing:

I = 0 = direct

I = 1 = indirect

The mode code specifies one of four modes of forming the address:

MM = 00 = Scratch area, 256 locations

MM = 01 = Relative to P forward

MM = 10 = Indexed

MM = 11 = Relative to P backward

ADD

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	M	M	I	A Field							

Adds contents of effective memory location to contents of A Register. Results stored in A.

Registers affected: A, OV

Timing: 2 plus 1 for each indirect level

SUB SUBTRACT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	M	M	I	A Field							

Subtracts the contents of effective memory location from contents of A Register. Results stored in A.

Registers affected: A, OV

Timing: 2 plus 1 for each indirect level

AND

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	M	M	I	A Field							

Performs the AND of the contents of the effective memory location and the contents of the A Register. Results stored in A.

Registers affected: A

Timing: 2 plus 1 for each indirect level

STA STORE A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	1	M	M	I	A Field							

Stores contents of the A Register into the effective memory location. A is unchanged and previous contents of memory are lost.

Registers affected: Memory

Timing: 2 plus 1 for each indirect level

STX **STORE X**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	1	M	M	I	A Field							

Stores contents of the X Register into the effective memory location. X is unchanged and the previous contents of memory are lost

Registers affected: Memory

Timing: 2 plus 1
for each indirect level

LDA **LOAD A**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	0	M	M	I	A Field							

Loads the contents of the effective memory location into the A Register. Memory is unchanged.

Registers affected: A

Timing: 2 plus 1
for each indirect level

LDX **LOAD X**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	M	M	I	A Field							

Loads the contents of the effective memory location into the X Register. Memory is unchanged.

Register affected: X

Timing: 2 plus 1
for each indirect level

IOR INCLUSIVE OR

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	0	M	M	I	A Field							

Inclusively OR's the contents of the effective memory location with contents of the A Register. Memory is unchanged.

Registers affected: A

Timing: 2 plus 1 for each indirect level

XOR EXCLUSIVE OR

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	M	M	I	A Field							

Performs the Exclusive OR of the contents of the effective memory location and the A Register. Memory is unchanged.

Registers affected: A

Timing: 2 plus 1 for each indirect level

EMA EXCHANGE MEMORY AND A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	1	M	M	I	A Field							

Simultaneously stores contents of A Register in the effective memory location and loads contents of effective memory location into the A Register.

Registers affected: A and Memory

Timing: 2 plus 1 for each indirect level

IMS INCREMENT MEMORY AND SKIP ON ZERO RESULT

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	0	1	1	M	M	I	A Field

The contents of effective memory location are incremented by one count and replaced. If the incrementing causes the result to become zero, a one place skip occurs. Overflow is set if the result of the incremention is 100000_8. NOTE: Neither skip nor OV occurs if IMS is executed as an interrupt instruction.

Registers affected: OV, Memory and P

Timing: 2 plus 1 for each indirect level

JMP JUMP UNCONDITIONAL

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	1	1	0	M	M	I	A Field

The A Field is placed in the P counter if the Jump is direct to scratchpad. If relative to P is specified, the A Field is added (or subtracted) to P and the results placed in P. If indexing is specified, the A Field is added to the contents of the X Register and the results placed in P.

If indirect addressing is specified the Jump occurs after the last level of addressing.

Registers affected: P

Timing: 1 plus 1 for each indirect level

JST JUMP AND STORE

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	1	1	1	M	M	I	A Field

The contents of the P counter (P + 1) are stored in the effective memory address. The P counter is changed after the store to contain the effective memory address plus one. The effective memory address is obtained in the same manner as in any other Memory Reference instruction.

Registers affected: P and Memory

Timing: 2 plus 1 for each indirect level

CMS COMPARE AND SKIP IF HIGH OR EQUAL

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	0	M	M	I	A Field							

Compares contents of effective memory location with contents of A Register and tests for A equal to, less than or greater than memory.

- If A less than memory, next instruction in sequence is executed (no skip).
- If A greater than memory, a one-place skip occurs.
- If A equal to memory, a two-place skip occurs.

Registers affected: P

Timing: 2 plus 1 for each indirect level

SCN SCAN MEMORY

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	1	1	0	1	A Field							

Scans table in memory specified by X Register and base address. Compares memory to contents of A Register (Key). Contents of X specifies the number of words in table to be scanned. The base address (minus one) of the table is stored as an indirect pointer in the scratch area, and the first location is accessed by indirect indexed addressing. If a comparison is found (A equal to memory), a one-place skip occurs. If A and memory are not equal, the X Register is decremented, M Register is decremented and the next sequential memory location is compared. If X goes to zero, the next instruction is executed. Since X is kept current, a return to the table to pick up the scan after a comparison is accomplished by executing the SCN instruction again.

- If OV is reset, a comparison is made against the full contents of the A Register. That is, all 16 bits are compared.
- If OV is set, a comparison of the upper 8 bits of the A Register is made. The lower eight bits are ignored.

Registers affected: X, M, P

Timing: 2 plus 1 for each indirect level plus 1 for each additional compare.

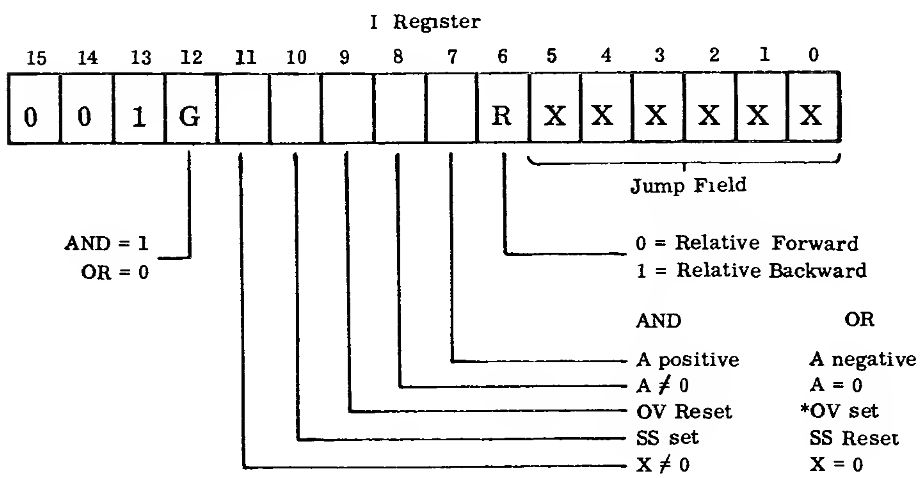
CONDITIONAL JUMP INSTRUCTIONS

Instructions that test conditions within the controller and take action depending upon the results of the test fall into the Conditional Jump instruction class. If the condition tested is satisfied, a jump of 1 to 64 locations is executed by adding or subtracting the contents of the Jump Field (J field) to the Program Counter (P counter). If the condition is not satisfied, the next instruction in sequence is executed.

The Conditional Jump instructions provide for conditional branching within a program, or branching to another program. There are five items that can be tested by Conditional Jumps:

- Sign of A (positive or negative)
- Contents of A (zero or not zero)
- Contents of X (zero or not zero)
- OV -- set (1) or reset (0)
- Sense Switch on Console -- on (depressed) or off

CONDITIONAL JUMP INSTRUCTION FORMAT

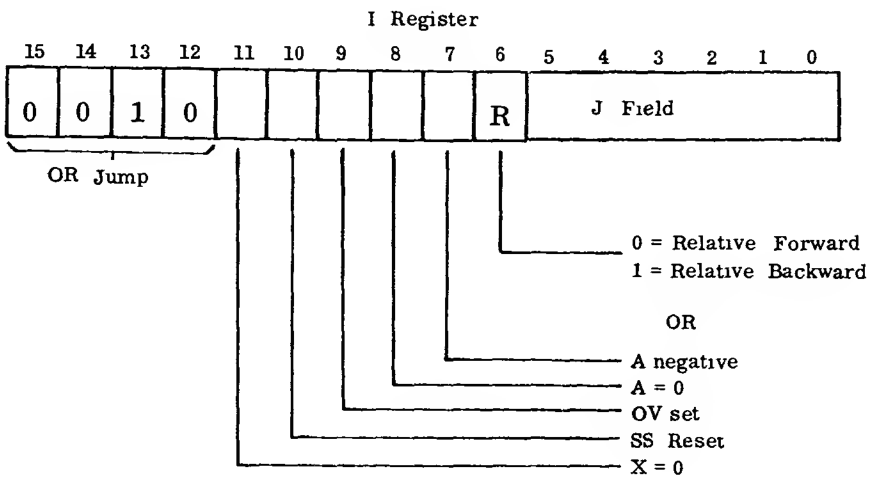


*NOTE: OV will be reset when tested for the set condition. If tested for the reset condition, it is unchanged.

Conditional Jump is a single-word, micro-programmed instruction. Bits 15-13 (001) specify the Conditional Jump class. Bit 12 specifies the AND (1) or OR (0) group of conditions. Conditions to be tested are selected by setting (to 1) the appropriate condition bits (11-7). Bit 6 is set (1) to produce a backward jump. The J field (bits 5-0), contains the relative jump address. A

forward jump is executed by adding the J field to the contents of the P register plus one. A backward jump is executed by subtracting the J field from the contents of the P register.

OR JUMP GROUP



OR Jump Instruction Format

The 31 instructions in this group are combinations of the five conditions that may be tested as shown above. If more than one condition is specified, the jump will occur if <u>any</u> of the specified conditions are met.

The following instructions are representative of the instructions that may be derived from this group. Refer to Appendix B for a complete listing of all conditional jump instructions.

JAM JUMP IF A MINUS

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	0	1	R	Jump Field					

A jump occurs if the A register is less than zero (A15 = 1). Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JAZ JUMP IF A ZERO

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	1	0	R	Jump Field					

A jump occurs if the A Register is zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JAL JUMP IF A LESS THAN OR EQUAL TO ZERO

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	1	1	R	Jump Field					

A jump occurs if the A Register is less than or equal to zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JOS JUMP IF OVERFLOW SET

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	1	0	0	R	Jump Field					

A jump occurs if the overflow bit is set (1) and the overflow bit is reset. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JSR JUMP IF SENSE SWITCH RESET

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	1	0	0	0	R	Jump Field					

A jump occurs if the SENSE SWITCH is reset (not depressed). Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JXZ JUMP IF X ZERO

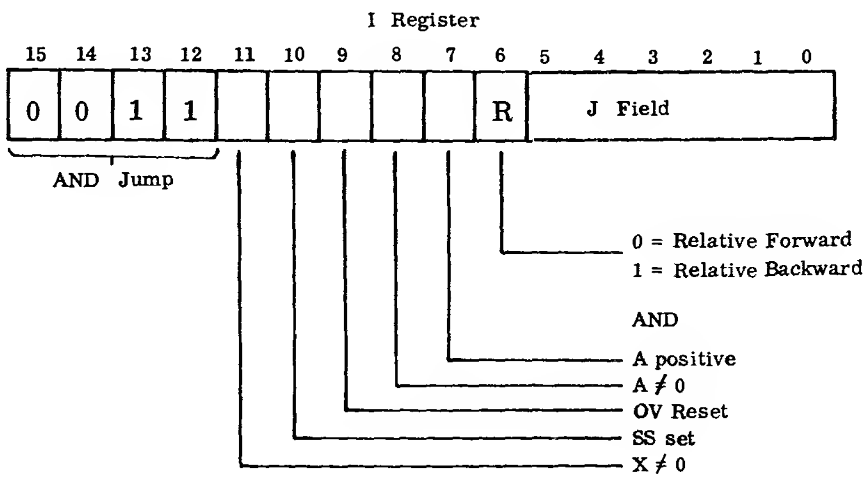
I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	0	1	0	0	0	0	R	Jump Field

A jump occurs if the X Register is zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

AND JUMP GROUP



AND Jump Instruction Format

The 31 instructions in this group are combinations of the five conditions that may be tested, as shown above. If more than one condition is specified, the jump will occur only if all of the specified conditions are met.

The following instructions are representative of the instructions that may be derived from this group. Refer to Appendix B for a complete listing of all conditional jump instructions.

JAP JUMP IF A POSITIVE

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	0	0	0	0	1	R	Jump Field

A jump occurs if the A Register is positive (A15 = 0). Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JAN JUMP IF A NOT ZERO

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	0	0	0	1	0	R	Jump Field

A jump occurs if the A Register is not zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JAG JUMP IF A GREATER THAN ZERO

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	0	0	0	1	1	R	Jump Field

A jump occurs if the A Register is greater than zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JOR JUMP IF OVERFLOW RESET

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	0	0	1	0	0	R	Jump Field

A jump occurs if the overflow bit is reset (0). Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JSS JUMP IF SENSE SWITCH SET

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	0	1	0	0	0	R	Jump Field

A jump occurs if the sense switch is set (depressed). Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

JXN JUMP IF X NOT ZERO

I Register

15	14	13	12	11	10	9	8	7	6	5 4 3 2 1 0
0	0	1	1	1	0	0	0	0	R	Jump Field

A jump occurs if the X Register is not zero. Otherwise the next instruction in sequence is executed.

Registers affected: P Timing: 1

IMMEDIATE INSTRUCTIONS

The immediate instructions allow certain operations that are similar to Memory Reference operations to be performed without going to memory for the operand. For example Add requires the operand to be located in memory, while Add Immediate, uses the last eight bits of the instruction as the operand.

The format of the Immediate Instructions is shown below:

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0				Operand							

Immediate Class   Op Code

IMMEDIATE INSTRUCTION FORMAT

The Immediate Instructions are limited to eight bit operands. When specifying a minus number, for instance Load A Immediate Minus, the eight bit operand is negated and all 1's are inserted in the upper half of the A register. Thus all 16 bits of A are affected.

CAI COMPARE to A IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	0	0	Operand							

The operand (lower half of instruction) is compared to lower half of A Register. If unequal a skip of one place occurs. If equal, the next instruction in sequence is executed. The contents of A are not disturbed.

Registers affected: P Timing: 1

CXI COMPARE to X IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	0	1	Operand							

The operand (lower half of instruction) is compared to lower half of X Register. If unequal, a skip of one place occurs. If equal, the next instruction in sequence

is executed. The contents of X are not disturbed.

Registers affected: P Timing: 1

AXI ADD to X IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	0	0	0	0	1	0	Operand

The operand (lower half of the instruction) is added to the lower half of the contents of the X Register. If a carry out of bit 7 occurs, it is added to the upper half of X (bit 8).

Registers affected: X, OV Timing: 1

SXI SUBTRACT from X IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	0	0	0	0	1	1	Operand

The operand (lower half of instruction) is subtracted from the lower half of the contents of X Register. If a borrow occurs, upper half of X is decremented.

Registers affected: X, OV Timing: 1

LXP LOAD X POSITIVE IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
1	1	0	0	0	1	0	0	Operand

The operand is loaded into the lower half of the X Register. The upper half is set to zero.

Registers affected: X Timing: 1

LXM LOAD X MINUS IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	1	0	1	Operand							

The operand is negated (2's complement) and loaded into the lower half of X Register. The upper half of X is set to all 1's.

Registers affected: X Timing: 1

LAP LOAD A POSITIVE IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	1	1	0	Operand							

The operand (lower half of instruction) is loaded into lower half of A Register. The upper half of A is set to zero.

Registers affected: A Timing: 1

LAM LOAD A MINUS IMMEDIATE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	1	1	1	Operand							

The operand (lower half of instruction) is negated (2's complement) and loaded into the lower half of the A Register. The upper half of A is set to all 1's.

Registers affected: A Timing: 1

REGISTER CHANGE INSTRUCTIONS

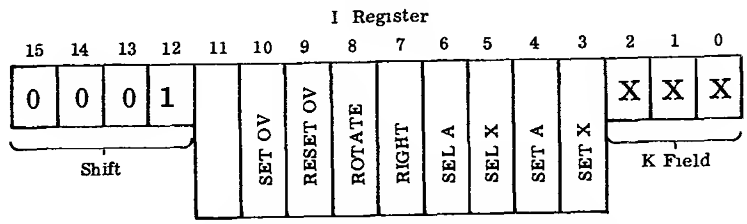
The Register Change class contain the instructions for handling the A and X Registers. The Shift instructions are a part of the Register Change class.

The instructions are micro-coded allowing many combinations, some of which are not useful except in special situations. The more useful ones are presented here and the bit assignments in the instruction are identified to allow the programmer to make up additional instructions.

Two micro codes are used; one for shift and one for the rest of the Register Change class.

SHIFT INSTRUCTIONS

The micro-code instruction format for the Shift instructions is shown below.



SHIFT INSTRUCTION FORMAT

<u>K field</u>. The K field is used to specify more than a one place shift. If K=0, a one place shift occurs. A maximum of eight places may be shifted in one instruction.

<u>SET X</u>. The Set X bit controls the set pulse to the X Register.

<u>SET A</u>. The Set A bit controls the set pulse to the A Register.

<u>SEL X</u>. The Sel X bit controls the select X logic. The contents of X are selected on to the S Bus with this bit on.

<u>SEL A</u>. The Sel A bit controls the select A logic. The contents of A are selected on to the S Bus with this bit on.

<u>RIGHT</u>. If this bit is on it indicates a right shift. Otherwise a left shift occurs.

<u>ROTATE</u>. The rotate bit controls the handling of the contents of the OV flip-flop. If the bit is on OV is shifted on to the register being shifted.

<u>RESET OV</u>. This bit controls the OV flip-flop. If the bit is on, OV is reset just prior to the shift. Thus a logical shift is created by resetting OV and shifting it in as in the rotate.

<u>SET OV</u>. This bit controls the set enable to the OV. It is not used for normal shifts.

ALA — ARITHMETIC SHIFT A LEFT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	0	1	0	1	0	X	X	X

The contents of the A Register (bits 0-14) are shifted left 1+K places. The sign bit (bit 15) is unchanged. Zeros are shifted into bit 0, and bit 14 is lost.

Registers affected: A Timing 1+1/4K

ALX — ARITHMETIC SHIFT X LEFT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	0	0	1	0	1	X	X	X

The contents of the X Register (bits 0-14) are shifted left 1+K places. The sign bit (bit 15) is unchanged. Zeros are shifted into bit 0, and bit 14 is lost.

Registers affected: X Timing: 1+1/4K

ARA — ARITHMETIC SHIFT A RIGHT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	1	1	0	1	0	X	X	X

The contents of the A Register are shifted right 1+K places. The sign bit (bit 15) is unchanged and propagated. Bit 0 is lost.

Registers affected: A Timing: 1+1/4K

ARX — ARITHMETIC SHIFT X RIGHT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	1	0	1	0	1	X	X	X

The contents of the X Register are shifted right 1+K places. The sign bit (bit 15) is unchanged and propagated. Bit 0 is lost.

Registers affected: X Timing: 1+1/4K

RRA ROTATE A RIGHT WITH OV

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	1	1	1	0	1	0	X	X	X

The contents of the A Register are shifted right 1+K places through the OV flip-flop. OV is shifted into bit 15.

Registers affected: A, OV Timing: 1+1/4K

RRX ROTATE X RIGHT WITH OV

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	1	1	0	1	0	1	X	X	X

The contents of the X Register are shifted right 1+K places through the OV flip-flop. OV is shifted into bit 15.

Registers affected: X, OV Timing: 1+1/4K

RLA ROTATE A LEFT WITH OV

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	1	0	1	0	1	0	X	X	X

The contents of the A Register are shifted left 1+K places through the OV flip-flop. OV is shifted into bit 0.

Registers affected: A, OV Timing: 1+1/4K

RLX ROTATE X LEFT WITH OV

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	1	0	0	1	0	1	X	X	X

The contents of the X Register are shifted left 1+K places through the OV flip-flop. OV is shifted into bit 0.

Registers affected: X, OV Timing: 1+1/4K

flip-flop. OV is shifted into bit 0.

Registers affected: X, OV Timing: 1+1/4K

LRA LOGICAL SHIFT A RIGHT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	1	1	1	0	1	0	X	X	X

The contents of the A Register are shifted right 1+K places through OV. Zeros are shifted into bit 15.

Registers affected: A, OV Timing: 1+1/4K

LRX LOGICAL SHIFT X RIGHT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	1	1	0	1	0	1	X	X	X

The contents of the X Register are shifted right 1+K places through OV. Zeros are shifted into bit 15.

Registers affected: X, OV Timing: 1+1/4K

LLA LOGICAL SHIFT A LEFT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	1	0	1	0	1	0	X	X	X

The contents of the A Register are shifted left 1+K places through OV. Zeros are shifted into bit 0.

Registers affected: A, OV Timing: 1+1/4K

LLX LOGICAL SHIFT X LEFT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	1	0	0	1	0	1	X	X	X

The contents of the X Register are shifted left 1+K places through OV. Zeros are shifted into bit 0.

Registers affected: X, OV Timing: 1+1/4K

LRR LONG ROTATE RIGHT *

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	1	1	0	0	0	X	X	X	X

Contents of A and X Registers are shifted right through OV 1+K places. OV is shifted into A15. X00 is shifted into OV.

Registers affected: A,X,OV Timing: 1+1/4K

LRL LONG ROTATE LEFT *

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	1	0	0	0	0	X	X	X	X

Contents of A and X Registers are shifted left through OV 1+K places. OV is shifted into X00. A15 is shifted into OV.

Registers affected: A, X, OV Timing: 1+1/4K

* NOTE: Long shift micro-coding deviates from the standard shift format. The K field is four bits, allowing up to 16-place shifts to be accomplished with one instruction.

ADDENDUM TO SHIFT INSTRUCTIONS

LLL LONG LOGICAL SHIFT LEFT*

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	1	1	0	0	0	0	X	X	X	X

$\underbrace{\text{bits } 3-0}_{\text{K Field}}$

The contents of the A and X Registers are logically shifted left through OV 1+K places. Zero is shifted into X00, X15 is shifted into A00, and A15 is shifted into OV. The previous contents of OV are lost. Up to 16 place shifts are allowed. A, X and OV act as a 33-bit register.

Registers affected: A, X and OV Timing: 1+1/4K

LLR LONG LOGICAL SHIFT RIGHT*

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	1	1	1	0	0	0	X	X	X	X

$\underbrace{\text{bits } 3-0}_{\text{K Field}}$

The contents of the A and X Registers are logically shifted right through OV 1+K places. Zero is shifted into A15, A00 is shifted into X15, and X00 is shifted into OV. The previous contents of OV are lost. Up to 16 place shifts are allowed. A, X and OV act as a 33-bit register.

Registers affected: A, X and OV Timing: 1+1/4K

NOR NORMALIZE X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	0	0	0	1	0	1	X	X	X

$\underbrace{\text{bits } 2-0}_{\text{K Field}}$

The contents of the X Register are arithmetically shifted left 1+K places or shifted until X15 is not equal to X14, whichever occurs first. Zero is shifted into X00. If X15 is not equal to X14, OV is set and the last shift is inhibited. That is, when X15 $\neq$ X14, the remaining shifts are inhibited and OV will be set to indicate the contents of X are normalized. Up to 8 place shifts are allowed.

Registers affected: X, OV Timing: 1+1/4K

* See Note page 2-20

MPS MULTIPLY STEP *

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	1	1	0	1	0	X	X	X	X

K Field

The MPS instruction is a combination ADD and Long Right Shift in which the contents of the R Register are conditionally added to A and the result shifted right. Both A and X are shifted. The instruction is used to code fast software multiply routines. Since the multiplicand is held in the R Register (which is not normally accessible to the programmer), care must be exercised in the use of MPS.

1 + K multiply steps are executed per the following algorithm:

1) If OV is set (contains a one), it is reset, and the contents of the R Register are added to the contents of the A Register. If an overflow occurred as a result of the addition, OV is set.

2) The A and X Registers are shifted right one place. A15 is shifted into A14, A00 goes to X15, X00 goes to OV, and the exclusive OR of OV and A15 goes to A15.

Registers affected: A, X, OV Timing: 1+1/4K

DVS DIVIDE STEP*

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	1	0	1	0	0	X	X	X	X

K Field

The DVS instruction is a combination ADD or SUB and Long Left Shift in which the contents of R are conditionally subtracted from or added to A and the result shifted left. Both A and X are shifted. The instruction is used to code fast divide subroutines. Since the divisor is held in the R Register (which is not normally available to the programmer), care must be exercised in the use of DVS.

1 + K divide steps are executed per the following algorithm:

1) The contents of the X Register are shifted left one place, with X15 going to a temporary bit store (LSRF) flip-flop.

2) X00 is set to zero if OV is not equal to R15, or it is set to one if OV and R15 are equal.

3a) If X00 is one, the contents of the R Register are subtracted from A and the result placed in A.

* See Note page 2-20

3b) If X00 is zero, the contents of the R Register are added to A and the result placed in A.

4) The contents of the A Register are shifted left one place with A15 going to OV and bit store (LSRF) going to A00.

Registers affected: A, X, OV Timing: 1+1/4K

NOTE: To load the R Register with an operand (multiplier or divisor), other registers are loaded normally and a Compare instruction executed. This will cause R to be loaded, but will not affect A or X.

Since the contents of R must be held throughout an MPS or DVS, and usually in multiply or divide subroutines, the routines should not be interruptable. This is accomplished by disabling interrupts at the beginning of the routines and enabling interrupts at the end of the routines.

REGISTER CHANGE

The micro-code instruction format for the Register Change instructions (excluding shifts) is shown below. Bit assignments that are different than the shift micro-code are described.

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	SEL $\overline{X}$	SEL $\overline{A}$	SEL R $\overline{R}$	$\overline{INCR}$	SEL A	SEL X	SET A	SET X	X	X	X
Register Change Class													K Field		

REGISTER CHANGE INSTRUCTION FORMAT

$\overline{INCR}$. Not Increment. When this bit is on, the carry into the adder is inhibited.

SELR$\overline{R}$. Select R and R Not. When this bit is on, both R and the 1's complement of R are selected onto the U-Bus to guarantee the U-Bus contains all zeros.

SEL$\overline{A}$. Select A Not. When this bit is on, the 1's complement of the contents of A Register is selected onto the S-Bus.

SEL$\overline{X}$. Select X Not. When this bit is on, the 1's complement of the contents of the X Register is selected onto the S-Bus.

NOTE: The S-Bus is an AND bus. Thus if the contents of A and X are both selected onto the S-Bus, the result is the AND of A and X on the S-Bus. If $\overline{A}$ and $\overline{X}$ are selected, the result is $\overline{A}$ AND $\overline{X}$ on the S-Bus. (Logically equivalent to $\overline{A + X}$.)

XRM SET X REGISTER to MINUS 1

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	1	X	X	X

Sets contents of X Register to all 1's.

Registers affected: A Timing: 1

ARM **SET A REGISTER to MINUS 1**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	1	0	X	X	X

Sets contents of A Register to all 1's.

Registers affected: A Timing: 1

AXM **SET A and X REGISTER to MINUS 1**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	1	1	X	X	X

Sets contents of A and X Registers to all 1's.

Registers affected: A, X Timing: 1

ZXR **ZERO X REGISTER**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	0	0	1	X	X	X

Sets contents of X Register to Zero.

Registers affected: X Timing: 1

ZAR **ZERO A REGISTER**

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	0	1	0	X	X	X

Sets contents of A Register to Zero.

Registers affected: A Timing: 1

ZAX ZERO A and X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	X	X	X

Sets contents of A and X Registers to Zero.

Registers affected: A, X Timing: 1

XRP SET X REGISTER to PLUS 1

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1	0	0	1	0	1	X	X	X

Sets contents of X Register to plus 1 (bit 0 on).

Registers affected: X Timing: 1

ARP SET A REGISTER to PLUS 1

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	1	0	1	0	X	X	X

Sets contents of A Register to plus 1 (bit 0 on).

Registers affected: A Timing: 1

AXP SET A and X REGISTERS to PLUS 1

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	1	0	1	1	X	X	X

Sets contents of A and X Registers to plus 1 (bit 0 on).

Registers affected: A, X Timing: 1

DXR DECREMENT X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	1	0	1	X	X	X

Subtracts one from the contents of X Register and places result in X.

Registers affected: X, OV Timing: 1

DAR DECREMENT A REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	1	0	1	0	X	X	X

Subtracts one from the contents of A Register and places result in A.

Registers affected: A, OV Timing: 1

IXR INCREMENT X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	1	0	1	X	X	X

Adds one to the contents of the X Register and places result in X.

Registers affected: X, OV Timing: 1

IAR INCREMENT A REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	1	0	1	0	X	X	X

Adds one to contents of A Register and places results in A.

Registers affected: A, OV Timing: 1

CXR COMPLEMENT X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	0	0	0	0	1	X	X	X

Performs 1's complement of contents of X Register and places result in X.

Registers affected: X Timing: 1

CAR COMPLEMENT A REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	0	1	0	X	X	X

Performs 1's complement of contents of A Register and places result in A.

Registers affected: A Timing: 1

NXR NEGATE X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1	0	0	0	0	1	X	X	X

Performs 2's complement of contents of X Register and places result in X.

Registers affected: X Timing: 1

NAR NEGATE A REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	0	0	1	0	X	X	X

Performs 2's complement of contents of A Register and places result in A.

Registers affected: A Timing: 1

TXA TRANSFER X to A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	1	1	0	X	X	X

Transfers contents of X Register to A Register. X is unchanged.

Registers affected: A Timing: 1

TAX TRANSFER A to X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	0	0	1	X	X	X

Transfers contents of A Register to the X Register. A is unchanged.

Registers affected: X Timing: 1

ANA AND of A and X to A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	1	1	0	X	X	X

AND's contents of A and X Registers and places result in A. X is unchanged.

Registers affected: A Timing: 1

ANX AND of A and X to X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	1	0	1	X	X	X

AND's contents of A and X Registers and places result in X. A is unchanged.

Registers affected: X Timing: 1

NRA NOR of A and X to A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	0	0	0	1	0	X	X	X

Performs NOR $(\overline{A + X})$ of contents of A and X Registers and places results in A. X is unchanged.

Registers affected: A Timing: 1

NRX NOR of A and X to X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	0	0	0	0	1	X	X	X

Performs NOR $(\overline{A + X})$ of contents of A and X Registers and places results in X. A is unchanged.

Registers affected: X Timing: 1

DAX DECREMENT A and Put in X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	1	0	0	1	X	X	X

Subtracts one from contents of A Register and places results in X. A is unchanged.

Registers affected: X, OV Timing: 1

DXA DECREMENT X and Put in A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	1	1	0	X	X	X

Subtracts one from contents of X Register and places results in A. X is unchanged.

Registers affected: A, OV Timing: 1

IAX INCREMENT A and Put in X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	1	0	0	1	X	X	X

Adds one to contents of A Register and puts results in X. A is unchanged.

Registers affected: X, OV Timing: 1

IXA INCREMENT X and Put in A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	1	1	0	X	X	X

Adds one to contents of X Register and places results into A. X is unchanged.

Registers affected: A, OV Timing: 1

CAX COMPLEMENT A and Put in X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	X	X	X

Places the 1's complement of contents of A Register into X. A is unchanged.

Registers affected: X Timing: 1

CXA COMPLEMENT X and Put in A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	0	0	0	1	0	X	X	X

Places the 1's complement of contents of X Register into A. X is unchanged.

Registers affected: A Timing: 1

NAX NEGATE A and Put in X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	0	0	0	1	X	X	X

Places the 2's complement of contents of A into X. A is unchanged.

Registers affected: X Timing: 1

NXA NEGATE X and Put in A

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1	0	0	0	1	0	X	X	X

Places the 2's complement of contents of X into A. X is unchanged.

Registers affected: A Timing: 1

ANB AND of A and X to A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	1	1	1	X	X	X

Places AND of A and X into A and X Registers.

Registers affected: A, X Timing: 1

NRB NOR of A and X to A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	0	0	0	1	1	X	X	X

Places NOR $(\overline{A + X})$ of A and X into both the A and X Registers.

Registers affected: A, X Timing: 1

DAB DECREMENT A and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	1	0	1	1	X	X	X

Subtracts one from contents of A and places results in both A and X.

Registers affected: A, X, OV Timing: 1

DXB DECREMENT X and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	1	1	1	X	X	X

Subtracts one from contents of X and places results in both A and X.

Registers affected: A, X, OV Timing: 1

IAB INCREMENT A and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	1	0	1	1	X	X	X

Adds one to contents of A and places results in both A and X.

Registers affected: A, X, OV Timing: 1

IXB INCREMENT X and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	1	1	1	X	X	X

Adds one to contents of X and places results in both A and X.

Registers affected: A, X, OV Timing: 1

CAB — COMPLEMENT A and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	0	1	1	X	X	X

Places the 1's complement of A in both A and X.

Registers affected: A, X — Timing: 1

CXB — COMPLEMENT X and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	0	0	0	1	1	X	X	X

Places the 1's complement of X in both A and X.

Registers affected: A, X — Timing: 1

NAB — NEGATE A and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	0	0	1	1	X	X	X

Places the 2's complement of A in both A and X.

Registers affected: A, X — Timing: 1

NXB — NEGATE X and Put in A and X

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1	0	0	0	1	1	X	X	X

Places the 2's complement of X in both A and X.

Registers affected: A, X — Timing: 1

CONTROL INSTRUCTIONS

The control instructions utilize various formats. NOP uses the Register Change format; OV control uses the Shift format and the other control instructions use a modified Register Change format which is shown below.

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	OP		SEL R $\bar{R}$	$\overline{INCR}$	SEL A	SEL X	SET A	SET X	X	X	X
													K Field		

00 = HLT
01 = EIN
10 = DIN

CONTROL INSTRUCTION FORMAT

Using the above format, several instructions can be formed which may or may not be useful. Only three of them are presented here. Others are left to the programmer's imagination.

NOP NO OPERATION

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	X	X	X	X	X	X	0	0	X	X	X

This instruction causes an 8 microsecond pause in the program. Bit locations marked with an X have no meaning in the instruction.

Registers affected: NONE Timing: 1

SOV SET OVERFLOW

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	1	0	0	0	0	0	0	0	X	X	X

Sets the Overflow flip-flop

Register affected: OV Timing: 1

ROV RESET OVERFLOW

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	0	0	0	0	0	0	X	X	X

Resets the Overflow flip-flop

Registers affected: OV Timing: 1

COV COMPLEMENT OVERFLOW

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	1	1	0	0	0	0	0	0	X	X	X

Complements the Overflow flip-flop

Registers affected: OV Timing: 1

EIN ENABLE INTERRUPTS

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	X	X	X	X	0	0	X	X	X

Sets the Enable Interrupt (ENIX) flip-flop in the processor.

Registers affected: None Timing: 1

DIN DISABLE INTERRUPTS

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	X	X	X	X	0	0	X	X	X

Resets the Enable Interrupt (ENIX) flip-flop in the processor. Prevents processor from responding to any interrupts.

Registers affected: None Timing: 1

HLT HALT

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	X	X	X	X	0	0	X	X	X

Halts the Controller.

Registers affected: None Timing: 1

III INPUT/OUTPUT SECTION

INTRODUCTION

The 816 Programmed Digital Controller is designed for communications, control, data acquisition and monitoring applications. Thus the I/O section of the unit has capabilities usually found only in much larger computers. The 29 I/O instructions provide the power and flexibility to allow the 816 to handle tasks other machines in its class cannot. Data transfers to and from the unit are either 8 or 16 bits parallel. The I/O bus utilizes the party-line technique to simplify the interface system. For real-time applications, the 816 has two fully implemented priority interrupt lines, plus a third interrupt request line that can be used to implement additional priority interrupts.

Interfacing to the 816 Controller is simple both logically and electrically. The use of standard DTL integrated circuits in the I/O eliminates special circuits and attendant complexity.

There are six types of I/O instructions: Sense, select, input to register (either A or X), output from register, input to memory or output from memory. By combining the sense with the input and output, two additional instructions are created: read to register and write from register. The Block Input/Output and the Load/Dump instructions allow great flexibility and high speed transfers to and from memory.

Each I/O instruction may refer to one of 32 different device numbers or addresses, and can operate up to eight functions per device address.

INPUT/OUTPUT INSTRUCTIONS

The I/O instructions allow the 816 Controller to communicate with the "outside world" — peripheral device controllers such as the teletype controller and special logic generated by Computer Automation or the user. Communication is in the form of commands which can trigger flip-flops or relays, sense signals which test the state of a flip-flop, relay, or incoming line, and data transfer operations which transfer parallel 8-bit bytes or 16-bit words into or out of the Controller.

All peripheral logic is connected in parallel on the I/O cable, which contains the data bus, control lines, and the device address bus. Selection of the device (peripheral logic group) with which communication is desired is accomplished by assigning each device a number or address and applying this number to the device address bus in the I/O cable during the I/O instruction. There are up to 32 device addresses available.

INSTRUCTION Format

The instruction format for I/O instruction is shown below. The first eight bits define the class, the op code within the class and the modifier for the op code. Of the remaining eight bits, five are used for the device address and three are used as a three bit function code. The function code allows up to eight functions to be created within the same device address for a given op code. For instance, at device 4, eight different flip-flops may be tested (sensed) by executing the "sense device 4" instruction eight times with the eight different function codes.

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1														
Class		Op Code			Mode			Device Address					Function Code		

I/O INSTRUCTION FORMAT

Sense Instructions

The sense and skip instruction allows the 816 to test the state of a specified function in an I/O device. Up to eight different conditions may be tested in each device. The modifier specifies whether the skip is to occur on a sense response from the device being tested or on no response.

SEN SENSE AND SKIP ON RESPONSE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	X	X	1								

The function specified will be tested in the device specified and a skip of one place will result if a response is obtained. If no response is obtained, the next instruction in sequence is executed.

Registers affected: P Timing: 1

SSN SENSE AND SKIP ON NO RESPONSE

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	X	X	0								

The function specified will be tested in the device specified and a skip will occur if a response is not obtained. If a response occurs, the next instruction in sequence is executed.

Registers affected: P Timing: 1

Select (External Control) Instruction

The select (SEL) instruction allows the 816 to pulse a specified flip-flop, relay, or external line in a device controller or special logic connected to the I/O bus. Instructions such as start tape, rewind, clear, stop reader, are examples of uses of the SEL instruction. Up to eight control functions may be implemented for each device address.

SEL SELECT FUNCTION

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	X	X	X								
Class		Op Code			Mode			Device Address					Function Code		

The control function specified by the function code is executed.

Registers affected: None Timing: 1

Input to Register Instructions

There are 16 I/O instructions that can cause data to be transferred from I/O devices to the A or X registers in the 816 processor. All register input instructions operate either with A or X.

Briefly the types of input to register instructions are:

- input 16-bit word unconditionally
- input byte unconditionally
- input 16-bit word conditioned on sense response

- input byte conditioned on sense response
- input 16-bit word, masked, unconditionally
- input byte, masked, unconditionally
- input 16-bit word, masked conditioned on sense response
- input byte, masked, conditioned on sense response

Masking is accomplished by performing the AND of the data in the register to be loaded with the data coming in from the I/O.

The transfer instructions that are conditioned on a sense response (for instance Read to A Register) will first test or sense the device to determine if the device is ready to make a transfer. If the response is obtained, the transfer is made. If the response is not obtained, the instruction is executed again. Thus the Read instruction will test the device once each eight microseconds until the sense response is obtained and the transfer made.

In situations where it is undesirable to have the processor wait on a peripheral, the separate sense, transfer and jump instructions can be used to perodically test the device and transfer when ready.

For byte inputs, the upper half of the register (either A or X) is undisturbed. This allows transferring a byte into the lower half of the register, shifting it left eight places and bringing in another byte for packing two bytes per word.

The formats shown for the input instructions contain an R in bit 9 which is the bit that specifies whether the A or X register is to be used:

R = 0 = A Register

R = 1 = X Register

INA INPUT TO A REGISTER (UNCONDITIONALLY)
INX INPUT TO X REGISTER (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	0	R	0								
Class			Op Code			Mode				Device Address				Function Code	

A 16-bit word will be transferred from the device specified to the A or X Register. The function code can be used to designate the source of data within the device if multiple sources exist. Up to eight sources may be specified by the function code, allowing a 128-bit input data word to be handled in 8 successive inputs, greatly simplifying the device logic.

Registers affected: A or X Timing: 1

INAM MASKED INPUT TO A REGISTER (UNCONDITIONALLY)
INXM MASKED INPUT TO X REGISTER (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	1	R	0								

Class (15–14), Op Code (13–11), Mode (10–8), Device Address (7–3), Function Code (2–0)

The masked input instruction causes the input data word (16-bits) from the specified device to be AND'ed with the previous contents of the register and the results placed in the register. The source of the data word within the device may be specified by the function code if multiple sources exist.

Registers affected: A or X Timing: 1

RDA READ WORD TO A REGISTER
RDX READ WORD TO X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	0	R	1								

Class (15–14), Op Code (13–11), Mode (10–8), Device Address (7–3), Function Code (2–0)

The Read instruction first senses the specified function in the specified device. If a response is obtained, the transfer is made and the next instruction in sequence is executed. If no response is obtained, the P counter is decremented and the Read instruction is executed again. Thus the processor "hangs" on the Read instruction until the device responds.

Registers affected: A or X Timing: 1 minimum

RDAM READ WORD TO A REGISTER MASKED
RDXM READ WORD TO X REGISTER MASKED

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	1	R	1								

Class (15–14), Op Code (13–11), Mode (10–8), Device Address (7–3), Function Code (2–0)

This instruction is the same as RDA or RDX except the input is masked by previous contents of the selected register.

Registers affected: A or X Timing: 1 minimum

IBA INPUT BYTE TO A REGISTER (UNCONDITIONALLY)
IBX INPUT BYTE TO X REGISTER (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	R	0								
Class		Op Code			Mode			Device Address					Function Code		

An eight bit byte will be transferred from the specified device to the lower half (bits 0 through 7) of the selected register. The upper half of the selected register is undisturbed.

Registers affected: A or X Timing: 1

IBAM INPUT BYTE TO A REGISTER MASKED (UNCONDITIONALLY)
IBXM INPUT BYTE TO X REGISTER MASKED (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	R	0								
Class		Op Code			Mode			Device Address					Function Code		

The contents of the lower half of the selected register is AND'ed with the incoming data and the results placed in the lower half of the selected register. The upper half of the selected register is undisturbed.

Registers affected: A or X Timing: 1

RBA READ BYTE TO A REGISTER
RBX READ BYTE TO X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	R	1								
Class		Op Code			Mode			Device Address					Function Code		

The Read Byte instruction first senses the specified function in the specified device. If a response is obtained, the transfer is made and the next instruction in sequence is executed. If no response is obtained, the P counter is decremented and the Read Byte instruction is executed again. Thus the processor "hangs" on the instruction until the device responds. Only the lower half of the selected register is affected.

Registers affected: A or X Timing: 1 minimum

RBAM	READ BYTE TO A REGISTER MASKED
RBXM	READ BYTE TO X REGISTER MASKED

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	R	1								
Class		Op Code			Mode			Device Address					Function Code		

The instruction is a combination of the Read Byte and the masked Byte instructions. The processor "hangs" on the instruction until the device responds. When the response is obtained the contents of the lower half of the selected register are AND'ed with the incoming byte and the result placed in the lower half of the selected register. The upper half of the register is not affected.

Registers affected: A or X Timing: 1 minimum

Output from Register Instructions

The output register instructions transfer data from the A or X registers to the specified device. All 16 bits are presented to the device, but for byte oriented peripherals only the lower eights will be accepted. The contents of A and X are not disturbed.

OTA	OUTPUT A REGISTER (UNCONDITIONALLY)
OTX	OUTPUT X REGISTER (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	1	R	0								
Class		Op Code			Mode			Device Address					Function Code		

The contents of the A or X Register are transferred to the specified device. The function code can be used to specify various destinations within the device address if more than one destination exists. The contents of A and X are not altered.

Registers affected: I/O Timing: 1

OTZ — OUTPUT ZERO (UNCONDITIONALLY)

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	0	R	0								
Class		Op Code			Mode			Device Address					Function Code		

This instruction outputs all zeros on the data bus to the specified destination. A and X may have any value.

Registers affected: I/O — Timing: 1

WRA — WRITE FROM A REGISTER
WRX — WRITE FROM X REGISTER

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	1	R	1								
Class		Op Code			Mode			Device Address					Function Code		

The contents of A or X register are transferred to the specified device on a sense response. If no response is obtained, the P counter is decremented and the Write instruction executed again. To prevent the data from being strobed into the device register the I/O signal PLSE is inhibited if no response is obtained. The processor "hangs" on the instruction until a response from the device is received. When the transfer is made, the next instruction in sequence is executed. The contents of A and X are not altered.

Registers affected: P, I/O — Timing: 1 minimum

WRZ — WRITE ZEROS

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	0	R	1								
Class		Op Code			Mode			Device Address					Function Code		

This instruction transfers zeros to the specified device upon response from the device. Contents of A and X are immaterial and are not altered.

Registers affected: I/O — Timing: 1 minimum

BLOCK TRANSFER INSTRUCTIONS

The block input/output instructions are powerful I/O instructions that allow access to memory without going through the A or X registers. Automatic data channels may be multiplexed with ease using the block I/O instructions as single execute interrupt instructions.

Three memory locations are required for the block instruction; instruction, word counter and address counter.

Four memory cycles are required for each input or output:

- Instruction cycle to fetch the instruction,
- Word cycle to fetch, increment and test the word counter,
- Address cycle to fetch and increment the address and transfer result to the memory address register,
- Operand cycle to transfer the data word to or from memory at the address specified by the address pointer.

The word counter and the address pointer are located immediately behind the block instruction:

P	Block I/O Instruction
P + 1	Word Counter
P + 2	Address Pointer

Block I/O Exits when used with interrupts

The word counter contains the 2's complement of the number of words to be transferred using the Block instruction. Each time the Block instruction is executed, the word counter is incremented by one and tested for zero. If after incrementing it is zero, a flag is set within the processor indicating this is the last transfer (end-of-block). During the transfer cycle this flag will cause an echo pulse to be sent to the device transferring data.

NOTE: The block may be any size, limited only by memory available.

The address pointer contains one less than the next memory location. After the word counter has been accessed, the address pointer is incremented, replaced and the incremented value used as the operand address. Any address may be used as the starting address. This can be dangerous if care is not taken to insure the proper starting address since the scratch area can be reached by the Block instructions. It is also a very useful feature since all of memory can be used by the Block instructions.

The most powerful use of the Block instructions is as interrupt instructions. Since Block does not alter A, X, P, or OV when executed by interrupts, automatic data channels may be implemented with ease. As many channels as there are interrupt lines may be multiplexed since each Block instruction carries its own word counter and address pointer with it. The end-of-block (EOB) Echo notifies the interrupting peripheral when the last transfer is made and this action can be used to cause an EOB interrupt from that peripheral. The EOB interrupt may drive its own interrupt line or it may share the IURX line with other devices and the device interrupting with an EOB interrupt may be determined by polling.

When the Block instructions are not used with interrupts, the instruction following the address pointer is the EOB exit and will be executed when the last word in the block has been transferred. The instruction following the EOB exit will be executed if the end of the block has not been reached.

P	Block Instruction
P + 1	Word Counter
P + 2	Address Pointer
P + 3	EOB Exit (usually jump)
P + 4	Next Instruction

Block I/O Exits when not used with interrupts

INB INPUT BLOCK TO MEMORY

<table>
<tr><td colspan="16">I Register</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="2">Class</td><td colspan="3">Op Code</td><td colspan="3">Mode</td><td colspan="5">Device Address</td><td colspan="3">Function Code</td></tr>
</table>

The Block Input instruction transfers one word from the specified device to the address specified by the address pointer (after the pointer is incremented). The word counter is incremented and tested for zero result. The word counter and the address pointer must be located directly after the instruction. The instruction acts somewhat like a "data break" or "cycle steal" when used with interrupts. The operating program will see a 32μs pause for each transfer.

Registers affected: Memory Timing: 4

OTB OUTPUT BLOCK FROM MEMORY

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	0								
Class		Op Code			Mode			Device Address					Function Code		

The Block Output instruction transfers one word to the specified device from the address specified by the address pointer (after the pointer has been incremented). The word counter is incremented and tested for zero result. The word counter and the address pointer must be located directly after the instruction. The instruction acts somewhat like a "data break" or "cycle steal" when used with interrupts. The operating program will see a 32μs pause for each transfer.

Registers affected: I/O Timing: 4

LOAD/DUMP MEMORY INSTRUCTIONS

To allow very high speed input and output from memory, the 816 incorporates a very powerful pair of instructions called LOAD MEMORY and DUMP MEMORY. A block of any length (limited to maximum memory size) may be transferred into or out of memory at a maximum rate of 125,000 words (16-bits) per second. The processor is totally devoted to the instruction until the entire block has been transferred. Data does not go through the A or X Registers, but rather goes directly to or from memory.

The operation of the instruction is as follows. The block length (number of words to be transferred) is placed in the X Register. Each transfer causes the X Register to be decremented and when the contents of X equals zero the instruction terminates. The base address minus 1 of the block is stored in the next memory location following the LOAD/DUMP instruction. The first transfer is made at the address formed by adding X to the base address pointer. Thereafter the memory address register is decremented after each transfer. Essentially, the first location is reached by indexed indirect addressing.

The transfer of data is made only on response from the peripheral device. If no response is obtained, X and M are not decremented and the device is tested again (this requires another memory cycle). Thus the input rate is determined by the speed of the peripheral device. The maximum rate is 125,000 words/second or 8μs per transfer. If the maximum rate is not achieved by the peripheral the next slower rate is 16μs per transfer, then 24μs and so on. The processor will test every 8μs and make transfers on those cycles where a response is obtained.

After the last transfer in the block is made, the next instruction in the program is executed.

The LOAD/DUMP instructions cannot be interrupted. Thus in real time systems consideration should be given to the amount of delay allowable in responding to interrupts and the block lengths kept to a size to accommodate the interrupts.

The LOAD/DUMP instructions cannot be used as a normal interrupt instruction since the X and P Registers are altered by the instruction.

No special I/O logic is required in the peripheral control logic to utilize the LOAD/DUMP instructions.

LDM LOAD MEMORY

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	0	0	X	1								
Class		Op Code			Mode			Device Address					Function Code		

This instruction loads a block of data from the specified device upon a sense response from the condition specified by the function code. The memory location immediately following the instruction must contain the base address minus one. The X Register must contain the number of words in the block to be transferred. The first data word will be stored in location "X plus base address pointer". The last word in the block will be stored in location "base address", i. e., the table is loaded backwards. After the last word is transferred, the instruction following the base address pointer is executed.

Registers affected: Memory, X Timing: Indefinite

DPM DUMP MEMORY

I Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	0	1	X	1								
Class		Op Code			Mode			Device Address					Function Code		

This instruction operates exactly as the LOAD instruction except data is transferred from memory to the peripheral.

Registers affected: I/O, X Timing: Indefinite

PRIORITY INTERRUPT SYSTEM

The 816 priority interrupt system makes the controller an extremely flexible and powerful real-time systems component. The interrupt system provides:

- Ability of fast response to external stimuli,

- Ability to automatically transfer blocks of data without attention of operating program,
- Ability to multiplex data to and from memory automatically,
- Ability to generate timing intervals and or time-of-day clock (with Real Time Clock option),
- Ability to react to low power line conditions (using Power Fail/ Restart option).

When an interrupt request is received by the controller, the controller responds to the interrupt at the end of the current instruction being processed (provided the interrupts are enabled). Control is transferred to the memory location associated with the particular interrupt line requesting the interrupt. There are four reserved memory locations reserved for each interrupt line in the system. Normally only one location is required, but when Block I/O is used as an interrupt instruction, two adjacent locations are required. Since it is easier to address every fourth location than every third one, the interrupt instruction locations are four locations apart.

For single-instruction interrupts, the interrupt instruction is one of the 816 instructions that does not change the program counter, the A Register or the X Register. A single-instruction interrupt can be serviced without altering the operating program except to delay it for the amount of time it takes to execute the interrupt instruction.

The most common interrupt instruction is Jump and Save which transfers control to a sub-routine designated by the Jump and Save instruction. The contents of P are saved (stored) in the first memory location of the sub-routine and is used later for a return jump back to the operating program. The number of interrupt levels available with the 816 is limited only by economics. Interrupt lines are added in groups of eight and using every fourth memory word, up to 1,000 interrupts could be put in a 4K system.

Interrupt addressing is accomplished two ways; hard-wired addresses in the processor for the standard interrupt lines, and externally supplied addresses for interrupts using the IURX line.

- Interrupt lines 1 and 2 (IL1X and IL2X) have address generators in the processor to force the unit to go to locations 2 and 6 respectively.
- For all other interrupts, the processor requests an interrupt address from the interrupting device with the signal IARX. The interrupt address is placed on the data bus by the interrupting peripheral and it is then transferred to the M Register. If the interrupt instruction involves pointers as in the case of Block I/O, the M Register is incremented to obtain their addresses.

Priority of interrupts are determined by the processor logic and/or system wiring among the interrupt devices. The highest priority is given to the Power Fail/Restart option (when it is in the system) followed by Interrupt Line 1 (IL1X) and Interrupt Line 2 (IL2X). The assignment of priority among the

remaining interrupt devices is determined by the wiring of the Priority Out (PROT) signal which emanate from the processor when none of the three highest interrupts are present.

Control of the interrupt system is implemented with the Enable Interrupts mask flip-flop ENIX. When it is off, the 816 will not respond to any interrupt requests regardless of priority. The flip-flop is automatically turned off when a Jump and Save instruction is executed under interrupt. This allows time to disarm the line causing the interrupt before enabling other interrupts. Each interrupt line driver in peripheral logic has an individual arm/disarm mask associated with it to allow selective enabling or selective disabling.

Two control instructions allow the ENIX flip-flop to be set or reset under program control.

Single-instruction interrupts do not require control of the interrupt mask by the program. For example, if Teletype Buffer Ready flip-flop is the stimuli causing an interrupt and the interrupt instruction is the Block Output instruction, the stimuli is removed by the time the interrupt instruction is over. If the interrupt instruction had been Jump and Save, the interrupt request would remain and would cause another interrupt before the transfer to teletype has occured, with the result that the return address is within the interrupt subroutine and the controller cannot recover from the loop. Thus it is necessary to automatically disarm the interrupt system until one of two things occur; either (a) the line in the Teletype is disarmed or (b) the transfer to the Teletype occurs, which satisfies the interrupt condition.

Only certain 816 instructions are useful as interrupt instructions. For single-instruction interrupts, the following are most useful:

- Block Input
- Block Output
- Increment Memory (skip is prevented when Interrupt Acknowledge is on)
- Select

Instructions that alter A, X, OV or P registers are not generally useful as interrupt instructions since the interrupt may occur randomly within an operating program, and changing A, X or P randomly in an operating program can have disastrous results. (The Jump and Save instruction alters the P counter but its previous contents are saved.)

Interrupt location assignments are fixed for IL1X, IL2X, Power Fail/Restart, and Real Time Clock options.

0000	Restart interrupt instruction (Usually Jump to Restart subroutine)
0002	IL1X interrupt location
0006	IL2X interrupt location

0010 IURX interrupt location when Power Fail/Restart option is in the machine and no address is supplied by the interrupting device.

0018 Real Time Clock pulse interrupt location. (Usually Increment Memory)

001A Real Time Clock sync. interrupt location. (Usually Jump and Save)

001C Power Fail/Restart low power interrupt location. (Usually Jump and Save)

(NOTE: That the above locations are in hexedecimal.)

CONTROL CONSOLE (Refer to Appendix C)

The 816 Controller console consists of a sixteen-bit register display, sixteen data switches, four register select switches and miscellaneous control switches and indicators.

Register Display. A sixteen-bit register display is provided to allow viewing the contents of a selected register. Contents of A, X, I and P may be displayed while the controller is halted. In the RUN mode the contents of the A Bus are displayed.

Data Entry Switches. Sixteen data entry switches are provided for entering data into the selected register. The switches are momentary action. Data is entered immediately upon depression if the controller is halted and in the STEP mode. In the RUN mode, the switches are disabled.

Register Select Switches. Four interlocked, alternate action switches are provided for selection of the register to be connected to the display and the data switches. The switches are wired on a priority basis such that should two switches be depressed at one time, the switch on the left takes priority, disabling the one on the right. This prevents the contents of the two registers from being mixed and placed in both registers.

CLEAR Switch. A momentary action switch is provided for clearing the selected register. The switch is disabled in the RUN mode.

CYCLE Switch. A momentary action switch is provided for cycling the controller in the STEP mode, and for initiating the RUN mode if the STEP/RUN switch is in the RUN (out) position.

STEP/RUN Switch. An alternate action switch is provided for controlling the two operating modes of the controller. When the switch is out the controller will enter the RUN mode when the CYCLE switch is depressed. If the controller is in the RUN mode, depressing the STEP/RUN switch will cause the controller to enter the STEP mode; i.e., the controller will halt at the end of the next instruction.

STEP Indicator. An indicator is provided to indicate the controller is in the STEP mode (controller halted and the STEP/RUN switch in the STEP (in) position).

RUN Indicator. An indicator is provided to indicate the controller is in the RUN mode.

OVERFLOW (OV) Indicator. An indicator is provided to indicate the state of the Overflow flip-flop. The indicator lights when the OV flip-flop is set.

MANUAL EXECUTE Switch. An alternate action switch is provided to allow the operator to insert an instruction into the I Register and manually execute it by depressing the CYCLE switch.

SENSE Switch. An alternate action switch is provided to allow the operator to interface to an operating program. The state of the SENSE switch can be tested under program control, allowing branching conditions to be executed conditioned on the state of the SENSE switch. The switch may be changed in the RUN mode.

MEMORY DISABLE. An alternate action switch is provided to allow the operator to disable memory. When the MEMORY DISABLE switch is depressed, the current generators in the memory core stack drive circuitry are disabled, inhibiting the memory from operating. This allows contents of memory to be protected from transients when power is collapsing on turn-off or when power is coming up on turn-on. If the controller is equipped with the Power Fail/Restart option, the MEMORY DISABLE switch need not be used except in instances where the power fail subroutine is not in memory.

> NOTE: The MEMORY DISABLE switch must not be depressed while the controller is in the RUN mode.

RESET Switch. A momentary action switch is provided to allow the controller system (including peripherals) to be initialized when power is turned on. The RESET switch initializes the processor flip-flops, placing the controller in the STEP mode, and provides a pulse on the I/O cable which is used by peripheral logic for initialization. The RESET switch may be used as a PANIC button to break a runaway loop.

POWER Switch. An alternate action switch is provided for controlling AC power to the power supply. Power is ON when the switch is depressed.

POWER Indicator. An indicator is provided to indicate when the POWER switch is depressed.

PERIPHERAL EQUIPMENT DESCRIPTION

This section describes some of the I/O devices that are available with the 816 Controller.

33 TELETYPE OPTION

The 33 Teletype Option provided the 816 system with four I/O features in one package: Keyboard input, page printer, paper tape reader and paper tape punch. The peripheral device is a Model 33TC Send-Receive set* operated in the duplex mode. A peripheral interface to connect the 816 to the teletype is contained in one option logic board and is mounted in the basic 816 chassis. The interface contains a buffer-shift register that performs parallel-to-serial conversion when outputing from the 816 to the teletype and serial-to-parallel conversion when inputing from the teletype to the 816. Additional control logic is used to implement external control functions and sense functions in the interface.

The teletype option allows printing and punching (output mode) at a rate of 10 char/sec. Paper tape can be read at a rate of 10 char/sec in continuous mode or one char at a time in the step mode.

Programming

The teletype option increases the number of useful I/O instructions in the 816 instruction set. Using 07 as the device address, the instructions associated with the teletype options are listed below.

4039 SELECT Keyboard. This instruction resets the Buffer Ready flip-flop and puts the teletype interface in the read mode.

403A STEP Read. This command causes the character under the read station on the paper tape reader to be read and the tape advanced one character. The reader switch on the teletype must be in the RUN position. The Buffer Ready flip-flop is reset.

403B SELECT Continuous Read. This command causes the paper tape reader to continuously read at a rate of 10 char/sec until the reader is stopped or the tape runs out. The reader switch must be in the RUN position. The Buffer Ready flip-flop is reset.

403C Initialize the teletype interface. This command resets the control flip-flops, stops the oscillator and puts the interface in a static marking condition. The Buffer Ready flip-flop is reset.

*The ASR-33 Teletype is manufactured by Teletype Corp.

403D	SET Word Xfer Mask. This command sets a mask flip-flop in the interface to enable an interrupt to be generated by Buffer Ready flip-flop. (The interrupt line is wired according to system requirements.)
403E	SET Block Xfer Mask. This command sets a mask flip-flop in the interface to allow an interrupt to be generated when the Word Xfer Mask is in the off state. The interrupt can be used to indicate "End of Block."
403F	RESET Masks. This instruction disables both interrupt lines in the teletype interface by resetting the mask flip-flops.
4939	Sense Buffer Ready. This instruction senses the On state of the Buffer Ready flip-flop, i.e., a true response will occur if the flip-flop is set.
493A	Sense Word Xfer Mask Off. This instruction senses the Word Xfer Mask flip-flop and generates a true response if the flip-flop is in the off state.
493B	Sense TTY not busy. This instruction senses the state of the TTY controller and generates a true response if the TTY is not printing or reading a character.
6838 thru 6F38	Output A or X Register to teletype. This instruction transfers the contents of the Register to the teletype interface and causes the character to be printed. If the punch is on, the character will also be punched.
6038	Output memory to teletype. This is a block output instruction that causes one word of a block to be transferred from the memory location specified by the output pointer (which the instruction automatically interrogates and updates). The character transferred is printed/punched on the teletype.
7938 thru 7F38	Input byte from teletype to the A Register. The character in the teletype interface buffer is transferred to the A Register. The word may be AND'ed with previous contents of the register.
5038	Input from teletype to memory. This is a block input instruction that inputs the word from the teletype interface buffer to memory. The memory location is specified by the input pointer.

Block transfers using the teletype as the I/O device provides the capability of reading or printing blocks interrupt control.

HI-SPEED READER OPTION

The Hi-Speed paper tape reader option (PTR) consists of a 300 char/sec optical reader and the interface logic between the reader and the 816. The reader is a unidirectional, eight-level unit that reads continuously or steps one character at a time under control of the interface. Each word (8-bit character) is read in parallel and transfered to a buffer register in the interface where it is held until it is transfered to the 816.

The reader mounts in a standard 19 inch equipment rack, requiring 7 inches of rack height. The interface is contained on one logic board and is mounted within the basic 816 chassis.

Programming

Adding the high speed paper tape reader to the 816 Controller increases the instruction set of the system. The instructions associated with the PTR are listed below.

4034	Initialize the reader interface. This instruction resets the control flip-flops in the interface, and makes the "buffer not ready."
4031	STOP Reader. This instruction causes the reader to stop.
4035	SET Word Xfer Mask (WXM). This instruction causes the WXM flip-flop to be set, enabling interrupts to be generated with Buffer Ready. (The actual use of interrupts is determined by system wiring.)
4033	RUN. This instruction causes the reader to continuously slew tape at 30 inches/sec.
4032	STEP READ. This instruction causes the reader to read one character and stop.
4036	SET Block Xfer Mask (BXM). This instruction causes the BXM flip-flop to be set enabling an interrupt to be generated when WXM is off, i.e., at end-of-block.
4037	Reset Masks. This instruction resets both the WXM and BXM flip-flops which disables interrupts for Buffer Ready and end-of-block.
4931	Sense Buffer Ready. This instruction tests the state of the Buffer Ready flip-flop and generates a true response if the flip-flop is set.
4932	Sense Word Xfer Mask. This instruction tests the state of the WXM flip-flop and generates a true response if the flip-flop is off.

7930 thru 7F30 Input Buffer to A or X Register (byte). This instruction transfers the contents of the buffer to the A or X Register.

5030 Input Buffer to Memory. This is a block input instruction that inputs the word from the buffer to memory. The memory location is specified by the contents of the input pointer.

HI-SPEED PUNCH OPTION

The Hi-Speed paper tape punch option (PTP) consists of a 60 char/sec. punch and the interface logic between the punch and the 816. The punch is an eight-level unit that can punch paper or mylar tapes.

The punch interface contains an 8-bit buffer register to hold information being punched, and the control and interface logic required to control the punch and communicate with the 816.

The punch mounts in a standard 19 inch equipment rack and requires 10-1/2 inches of rack height. The interface is contained on a logic board which mounts in the basic 816 chassis.

Programming

Adding the Hi-Speed paper tape punch to the 816 increases the instruction set of the system. The instructions associated with the PTP are listed below.

4034 Initialize. This instruction resets all control flip-flops in the punch interface and makes the buffer "not ready."

4030 Punch Contents of Buffer. This instruction is used when copying an existing paper tape.

4035 SET Word Xfer Mask (WXM). This instruction causes the WXM flip-flop to be set enabling an interrupt to be generated with Buffer Ready. (The actual use of interrupts is determined by system wiring.)

4036 SET Block Xfer Mask (BXM). This instruction causes the BXM flip-flop to be set enabling an interrupt to be generated when WXM is off, i.e., at end-of-block.

4037 Reset Masks. This instruction resets both the WXM and BXM flip-flops which disables interrupts for Buffer Ready and end-of-block.

4031 Sense Buffer Ready. This instruction tests the state of the Buffer Ready flip-flop and generates a true response if the flip-flop is set.

4832 Sense Word Xfer Mask. This instruction tests the state of the WXM flip-flop and generates a true response if the flip-flop is off.

6830 thru 6F30	Punch Contents of A or X Register. This instruction causes the contents of A or X Register to be transferred to the punch interface and punched.
6030	Output Memory and Punch. This instruction is a block output that outputs the word in memory specified by the output pointer. The word is transferred to the interface buffer and punched.

MAINFRAME OPTIONS

This section describes some of the options available with the 816 processor.

POWER FAIL/RESTART OPTION

The 816 Power Fail/Restart (PFR) option allows the controller to be operated from an unreliable AC source. A low power condition or a temporary power outage will be detected in time to allow the operating program to prepare for the power loss. When power returns to normal, the controller is automatically restarted. Unattended operation is possible at remote sites.

The PFR option consists of a voltage detector, interrupt register, and priority interrupt logic on an option module in the 816 chassis. Two interrupts are provided with the option. One interrupt flags the low power condition, and the other interrupt restarts the controller one second after power resumes.

Operation

The PFR option monitors the 115V AC line voltage and closes a switch when the voltage drops below a preset value (typically 100V). The switch closing sets a flip-flop which requests an interrupt. The interrupt location should contain a Jump and Save to the power fail subroutine. Approximately two milliseconds after low power is detected, inhibiting memory start pulses and shunting the current sources in the memory drive electronics prevent spurious memory cycles as power collapses.

When power returns, the detector output (indicating power is normal) is delayed one second and then used to create the restart interrupt. During the one second interval between power up and the restart of the controller, the 816 system is initialized. After the one second delay, the controller is forced to location zero to obtain the restart interrupt instruction (Jump to Restart subroutine).

Priority

The PFR priority is the highest in the interrupt system.

Reserved Memory Locations

The PFR option requires two memory locations for the two interrupt instructions.

001C	Power low interrupt instruction. (Jump & Save to shutdown routine.)
0000	Restart interrupt instruction. (Jump to Restart routine.)

REAL TIME CLOCK OPTION

The Real Time Clock (RTC) is a 816 processor option that provides a means of determining elapsed time and/or creating a time-of-day clock with software.

The RTC derives time pulses from the 60-cycle primary input to the 816. These time pulses are then used to generate an interrupt (clock interrupt) every 8.33 milliseconds to increment a counter in memory.

A second interrupt (sync interrupt) is generated by the RTC when the counter in memory reaches a count of 1,000 while being incremented by the clock interrupt instruction. The sync interrupt is used to enter a time keeping subroutine.

The two interrupts required by the clock and sync pulses are contained within the RTC. Interrupt addresses for each line are also generated by the RTC logic, providing a complete stand alone option.

Operation

The instruction set of the basic 816 system is increased by the RTC. The number of interrupt in the 816 system is also increased by two. Each instruction, including the suggested interrupt instructions, are described below.

Control Instructions

4040	Enable RTC. Sets a mask flip-flop in the RTC, allowing an interrupt to be requested by either pulse or sync (if sync is armed).
4042	Arm Sync. Sets the arm/disarm flip-flop on the sync interrupt, allowing an interrupt to be requested when clock counter in memory overflows.
4043	Clear RTC Interrupts. Resets both interrupt flip-flops (clock & sync) removing history from RTC. Does not disable or disarm RTC.
4044	Initialize RTC. Disarms, disables, and clears RTC preventing interrupts and removing history. Does not stop oscillator.
4047	Disarm Sync. Resets the arm/disarm flip-flop on the sync interrupt. RTC will store the sync pulse if received while sync interrupt disarmed.

Interrupt Instructions

Two memory locations in scratch area are reserved for the two interrupts associated with the RTC.

Location	Contents
0018	Increment Memory. RTC clock pulse interrupt instruction.
001A	Jump & Save. Sync interrupt instruction.

Priority

Priority of the RTC interrupts, relative to other interrupts in 816 system, is determined by the wiring of the PROT line. Since the two standard interrupt lines in the 816 have priority over all interrupts, (except PFR) the RTC competes with other interrupts in the 816 system for priority.

RESERVED MEMORY LOCATIONS

Since each interrupt line has a memory location associated with it, these locations must not be used for other purposes. As an aid to the programmer, all reserved locations in the basic 816 system plus mainframe options are listed. In larger systems additional locations may be required.

All reserved locations are in the scratch area.

LOC	USE
0000	Restart interrupt instruction
0002	Interrupt Line 1 instruction
0006	Interrupt Line 2 instruction
0010	IURX Polling instruction
0018	Real Time Clock pulse instruction
001A	Real Time Clock sync instruction
001C	Power Fail interrupt instruction

IV I/O INTERFACE REFERENCE

INTRODUCTION

The 816 Controller is a highly flexible systems component designed to be easily applied to communciations, control and monitoring tasks. Great care was taken to make the unit easy to program using assembly or machine language. The organization of the processor enables the 816 to obtain high memory efficiency, avoiding the problem of "core burning" that is so prevalent in small computers. Memory utilization is further enhanced by the powerful and flexible I/O instruction set.

The interface to the 816 is elegant in its simplicity. Considerable effort was expended to reduce the amount of interface logic required outside the controller. The interfacing "problems" have been solved inside the 816 leaving the user free to concentrate on his system rather than on a complicated and expensive interface. Standard DTL circuits are used throughout the I/O, providing additional savings in parts cost and power.

PARTY LINE I/O BUS

The 816 incorporates a "party-line" I/O bus to communciate with peripheral logic. Each peripheral logic group or "device" is given an address which is used in all communications between the 816 processor and that device. A block diagram of a 816 system with three devices is shown in Figure 4-1. Note that while the I/O cable is physically connected in a serial fashion, the devices are electrically connected in parallel.

A termination "shoe" containing line terminating resistors is connected to the end of the I/O cable at the last device. The resistor is required to maintain the integrity of the transmission line by terminating the line in its characteristic impedance (or approximation thereof) at the receiving end to minimize reflections. Since the Data lines are bilateral, i.e., they can be driven from either end, a termination is required at each end. The unilateral lines require a terminating resistor only on the receiving end. Figure 4-2 shows the circuit schematics of the two types of lines. All lines are twisted pair.

The length of the I/O cable should be kept as short as possible to minimize cross talk between lines. When possible peripheral logic should be located adjacent to the 816. The maximum length allowable will vary with the application, but in general it should be kept under ten feet.

The I/O cable consists of a number of lines grouped according to functions. Each functional grouping is called a bus. See Figure 4-9.

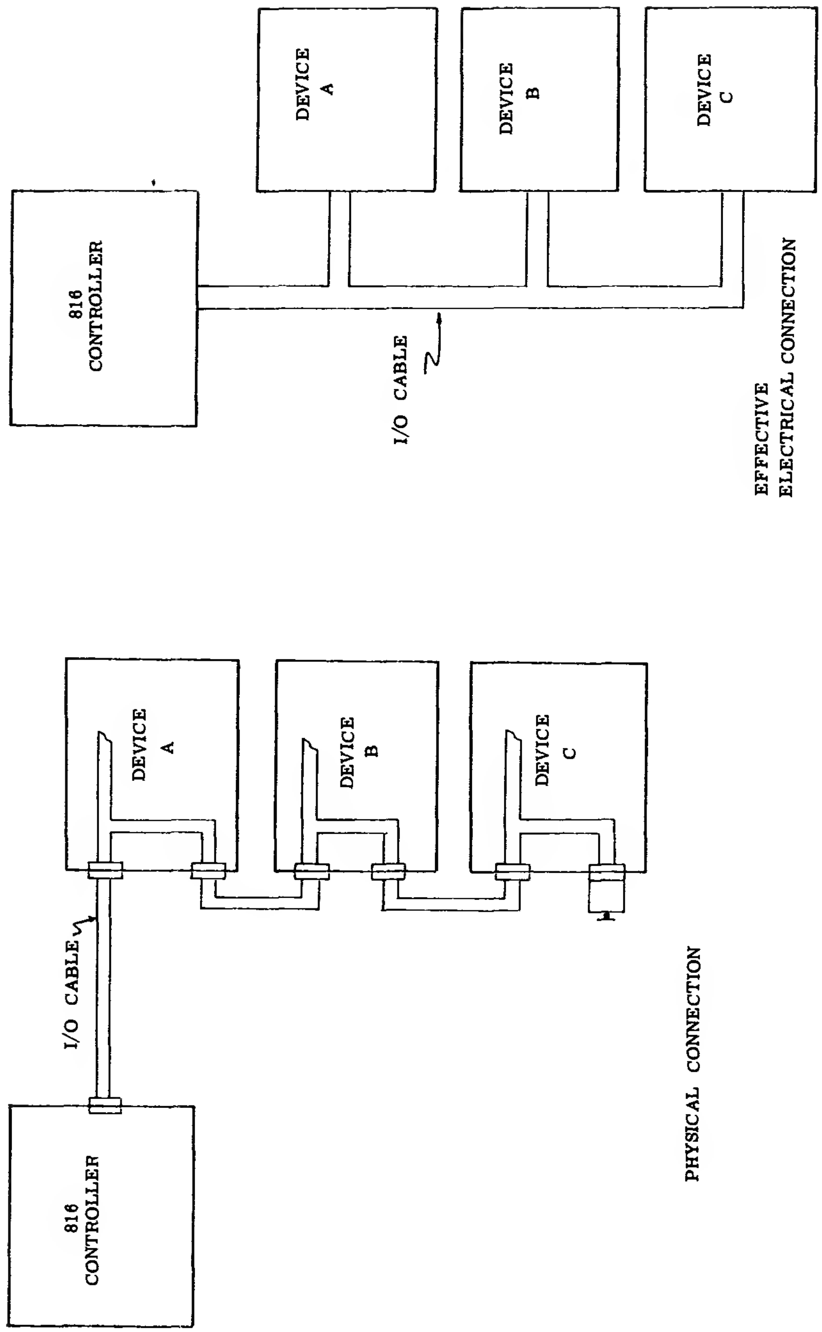


Figure 4-1 Block Diagram of 816 Controller System

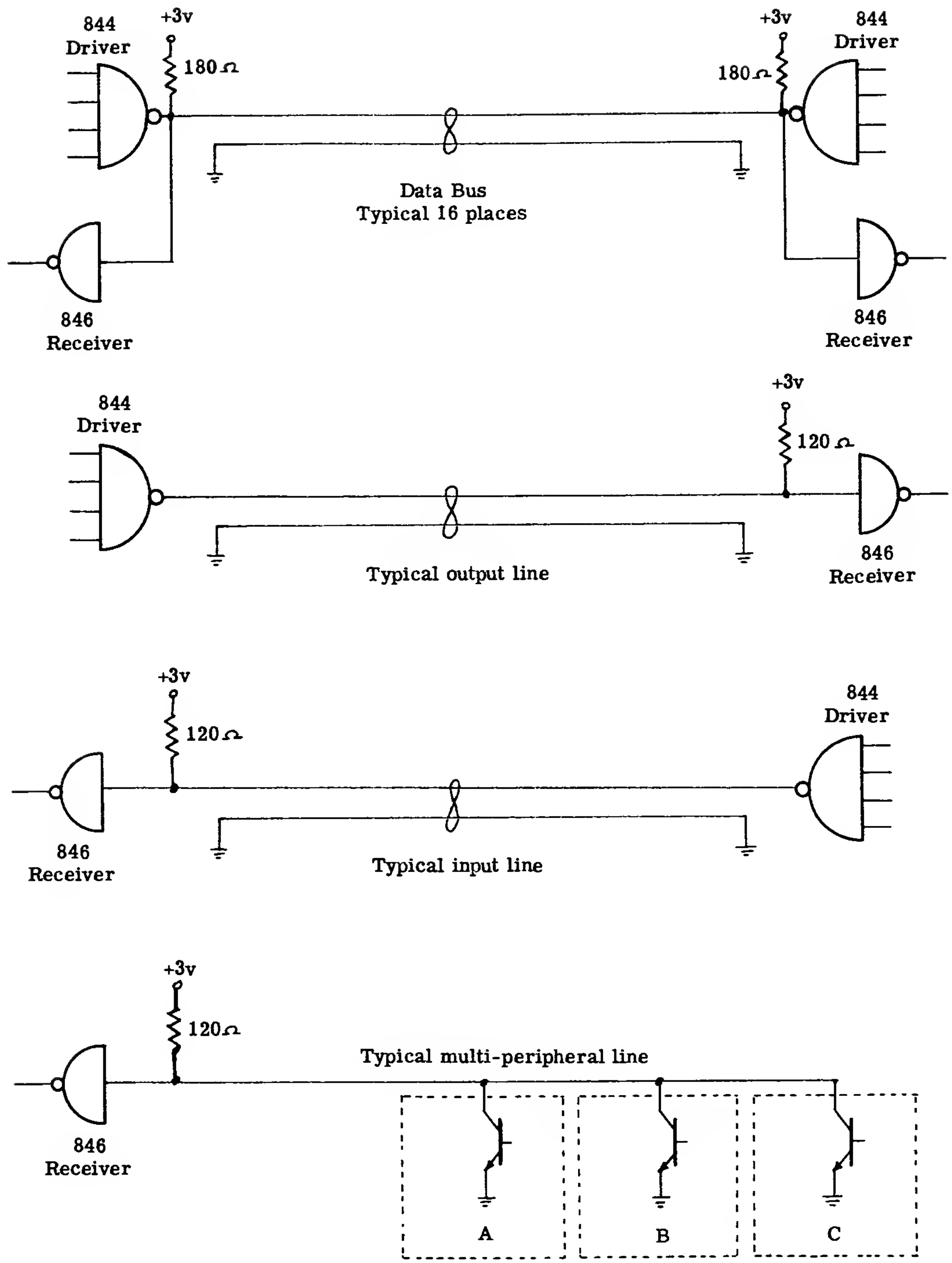


Figure 4-2 I/O Line Schematics

Data Bus — sixteen bidirectional lines that transmit data between devices and the 816. Each line is terminated on each end.

Peripheral Address Bus — five lines that transmit the device address to devices during the execution of an I/O instruction.

Function Bus — three lines that transmit the function code to the devices during an I/O instruction.

Control Bus — fourteen lines that transmit control signals between the 816 and I/O devices. Each signal is described below.

SELX	Select. Signal present for two microseconds during execution of Select instructions.
OUTX	Output. Signal present for two microseconds during execution of any output instructions.
INXX	Input. Signal present for two microseconds during execution of any input instructions.
PLSE	Pulse. Strobe signal present for 500 nanoseconds at the end of EXCX, INXX, OUTX and IARX. Used to create trigger pulses for control flip-flops and load pulses for strobing data into register.
ECHO	Signal generated by Processor during Block I/O instruction if the block pointer overflowed (became zero) when incremented. Signal is used to disable the Word Xfer interrupt at end-of-block.
RSTX	Reset. Signal generated when the Reset switch on the 816 console is depressed and during power fail restart. Signal is used to initialize the Processor and all I/O interface logic.
SERX	Sense Response. Signal that is present when an I/O interface is interrogated with Device Address and Function Code if the test or sense conditions are met. The SERX line can be true even though a Sense instruction is not being executed. Further gating is performed in the Processor.
IL1X	Interrupt Line 1. Priority Interrupt line with priority over all other interrupts (except PFR). When acknowledged, this interrupt forces Processor to memory location 0002 to obtain the interrupt instructions.
IL2X	Interrupt Line 2. Priority Interrupt line with priority next to IL1X. Interrupt address 0006 is reserved for this line.

IURX	Interrupt Request. Common request line used by Interrupt Module and other mainframe interrupts. Use of this line obtains the interrupt, but the interrupt address must be supplied by the device requesting interrupt during IARX time.
IARX	Interrupt Address Request. Signal that requests interrupt address from interrupting device at the beginning of Interrupt Acknowledge.
IUAX	Interrupt Acknowledge. Signal that indicates an interrupt instruction is being processed.
PROT	Priority Out. Signal that indicates neither IL1X, IL2X nor PFR are requesting interrupts and a downstream device may request interrupt. By serially chaining this line through priority devices, relative priorities between priority devices may be established.
IOCL	I/O Clock. A 500 K Hz pulse train for use with priority interrupt devices and I/O logic. The signal is interrupted during IUAX (Interrupt Acknowledge).

Miscellaneous Lines — the power lines for the termination resistors and several spare lines that are used for PROT wiring and other systems requirements.

NOTE: The signals appearing on the I/O cable are "ground true" logic, i.e., the false level is the high level. This permits the parallel connection of devices using the "wired OR" technique, and also conserves power since the line terminations do not dissipate power except when the true (ground) signal exists.

INTERFACE TIMING

The 816 employs an eight microsecond memory system. Consequently, considerable time is available in implementing the signals necessary to interface with peripheral logic, eliminating the need for fast or special circuits. The shortest pulse width used is the 500ns PLSE signal which is used as a strobe. All other signals are at least 2 microseconds in duration, providing plenty of time for decoding and gating.

OUTPUT COMMAND

The output instructions all employ the same peripheral logic and timing. Data are placed on the Data Bus for two microseconds and at the same time the device address and function code is placed on the Peripheral Address Bus and

Function Bus respectively. The OUTX control line is true for the 2 microseconds data is present indicating an output instruction is being executed.

At the end of the 2 μsecond OUTX signal, the PLSE signal occurs if the output transfer is to be made. In the case of a sense and output combination, the PLSE signal will not occur unless a sense response is obtained by the processor. The composite timing of an output command is shown in Figure 4-3.

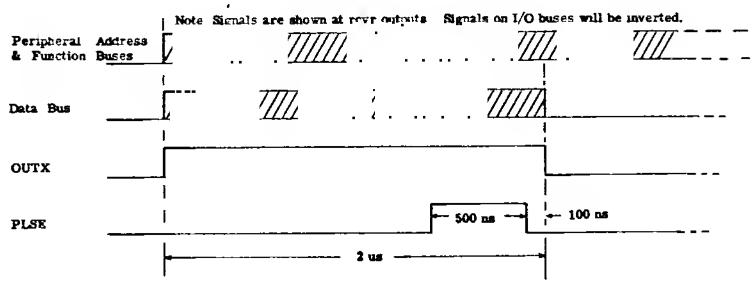


Figure 4-3 Output Timing

INPUT COMMAND

The input instructions all employ the same peripheral logic and timing. Data are placed on the Data Bus by the device which is addressed during the 2 microseconds the INXX signal is true. At the end of this time, the PLSE signal occurs if the input transfer is to be made. In the case of the sense and input instructions, the PLSE will not occur if a sense response is not obtained by the processor. The absence of the PLSE signal indicates the transfer was not made. The composite timing of an input command is shown in Figure 4-4.

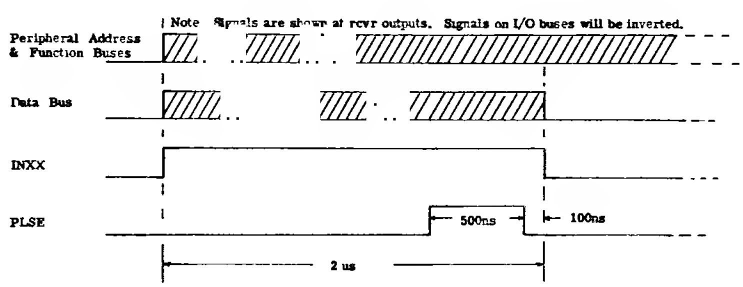


Figure 4-4 Input Timing

SELECT CONTROL COMMAND

The SEL instruction timing is similar to the transfer instructions except the Data Bus is not active. The SELX control signal is true for 2 microseconds.

The PLSE signal occurs at the end of the SELX signal and is used to generate set pulses which are gated by the Function Codes generated by decoding the Function Bus. Figure 4-5 shows the timing of the SEL command.

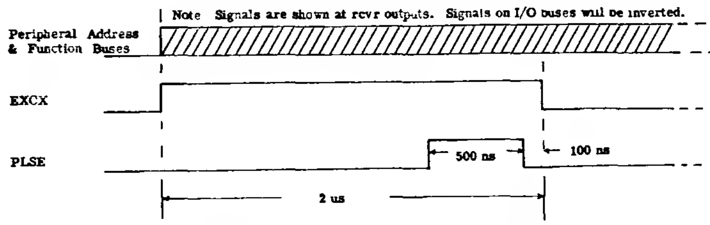


Figure 4-5 Select Control Timing

SENSE COMMAND

The sense instructions do not use a sense control line, since the final gating for sense is done inside the processor. This allows the sense instruction to be combined with the input and output instructions, creating powerful I/O instructions that conserve memory and shorten program execution times.

When a peripheral device is addressed by the processor, the function code is used to gate the condition of the specified function onto the sense response line, SERX. If a true response exists, the SERX line in the control Bus is driven to ground. Thus the sensing is accomplished during all I/O instructions, but using the results of the sense is determined by the instruction being executed.

The processor must receive the sense response one microsecond after the beginning of the instruction to allow time for decision making within the processor. Figure 4-6 shows the timing for Sense.

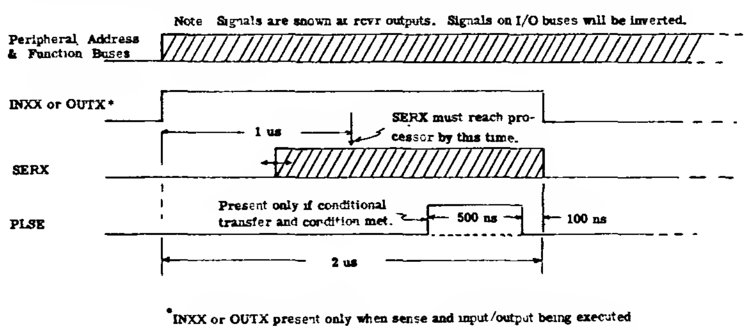


Figure 4-6 Sense Timing

INTERRUPT TIMING

The use of the standard interrupts in the basic 816 enables the controller to perform as a real time systems component responding to external stimuli and to allow considerable time savings in executing I/O programs that transfer data to/from peripheral devices.

A typical interrupt sequence using one of the two fully implemented interrupt lines in the 816 is as follows.

A. The Buffer Ready status flip-flop in a device turns on indicating that device desires data to be transferred from the 816 and drives the IL1X line in the Control bus to ground (true) state requesting an interrupt.

B. The processor finishes the current instruction and if interrupts are enabled, i.e., the processor is allowed to respond to interrupts, the request for interrupt will be acknowledged, and the instruction located in the reserved memory location associated with IL1X will be executed. The IUAX signal becomes true as soon as the request is acknowledged and stays on for the time required to execute the interrupt instruction, typically 8 to 32 microseconds.

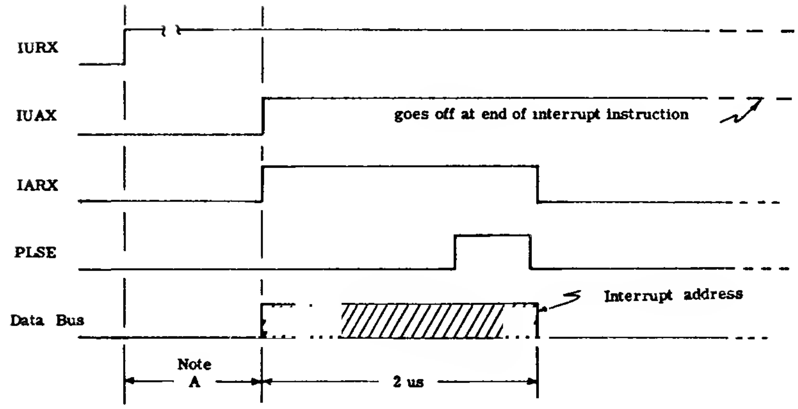
C. If the interrupt instruction is a JST, the interrupt enable flip-flop in the processor is turned off preventing further interrupts from being acknowledged until enable flip-flop is again turned on by the program.

D. A subroutine is entered (if JST used) which will transfer the desired data to the interrupting device, satisfying the request and resetting the Buffer Ready flip-flop which removes the interrupt request. Interrupts can now be enabled and a return Jump back to the operating program executed.

E. If the interrupt is a single-execute instruction such as Block Output, the request for data is satisfied by the execution of the interrupt instruction and the program continues as soon as IUAX goes off, creating only a 32 microsecond pause in the program.

The use of the IURX line is different from that of the IL1X and IL2X lines in that the interrupt address must be specified by the device requesting the interrupt whereas the IL1X and IL2X lines have associated address generators within the 816.

To obtain the interrupt address from the device, the processor sends an Interrupt Address Request signal IARX to the devices and the device requesting sends back the address the processor is to use to fetch the interrupt instruction. The IARX signal is 2 microseconds long. The PLSE signal occurs at the end of the IARX signal to notify the device the processor has received the address. The timing of the interrupt sequence using the IURX line in this fashion is shown in Figure 4-7.



Note A. Variable depending on instruction being executed.

Figure 4-7 Interrupt Timing

Another method of using the IURX line is the scanning technique. Several devices are allowed to generate an interrupt request at will and a subroutine is entered which polls each device capable of having generated a request. The polling is done by executing sense instructions until the requesting device is located. In this manner, priorities may be changed at will by the programmer by merely changing the order in which the devices are sensed. The technique is slower, however, than a fully automatic interrupt sequence but is considerably less expensive.

If a device requests an interrupt using IURX and does not respond with an address during IARX, the processor will access memory location 0000(Location 0010 if PFR option is used) for the interrupt instruction (which in almost all cases must be a JST).

All 816 standard peripheral interfaces have two interrupt line drives implemented. One driver is normally connected to IL1X and is used to generate an interrupt on Buffer Ready. The other driver is normally connected to IL2X and is used to generate an interrupt on end-of-block when performing block transfers under interrupt control. Each driver has a mask associated with it which may be controlled by the operating program.

Several peripheral devices may be connected to the IL1X and IL2X lines provided only one device is allowed to be operating under interrupt control at a given time. Since the IL1X and IL2X lines are fully implemented and therefore general purpose, they may also be used by customer interface logic without the need to generate priority and address logic outside the 816. This one fact alone typically <u>saves hundreds of dollars</u> over comparably priced machines in implementing real time systems interfaces.

Figure 4-8 shows the logic in a standard 816 peripheral controller that transfers data to/from the processor (Hi-speed reader, etc). The Buffer Ready flip-flop is set when the device is ready to send or receive data. If the Mask FF 1 is on, an interrupt is generated on IL1X which is used to cause data to be transfered. If the Block transfer instructions are used, an End-of-Block echo pulse occurs when the pointer overflows which resets Mask FF 1. This creates an interrupt on IL2X (if Mask FF 2 has been previously set) which is used to

signal the end of the block being transferred. Note that each interrupt line driver can be disabled by the mask flip-flops allowing complete system flexibility in the use of the interrupts.

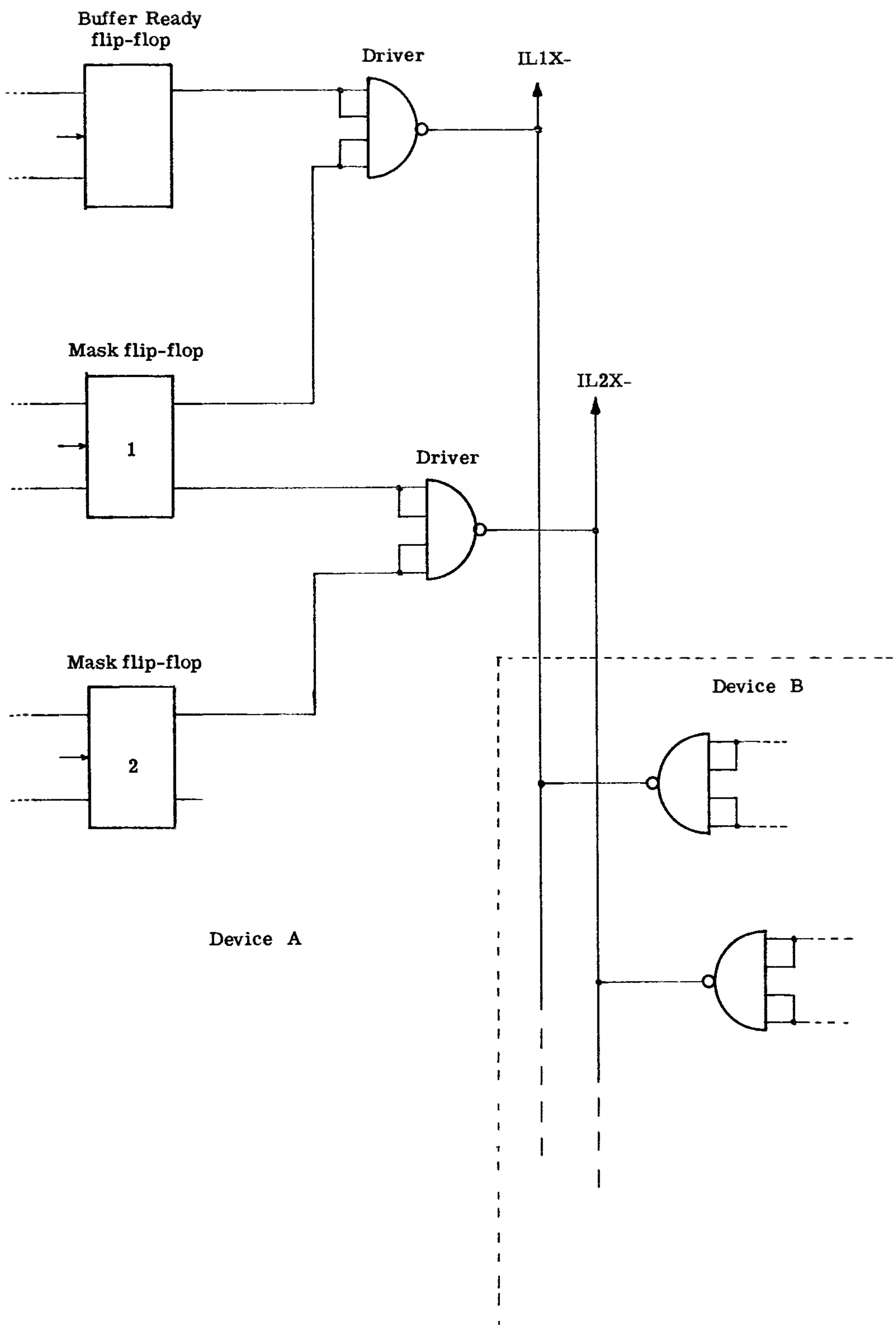


Figure 4-8 Standard Interrupts in Peripheral Devices

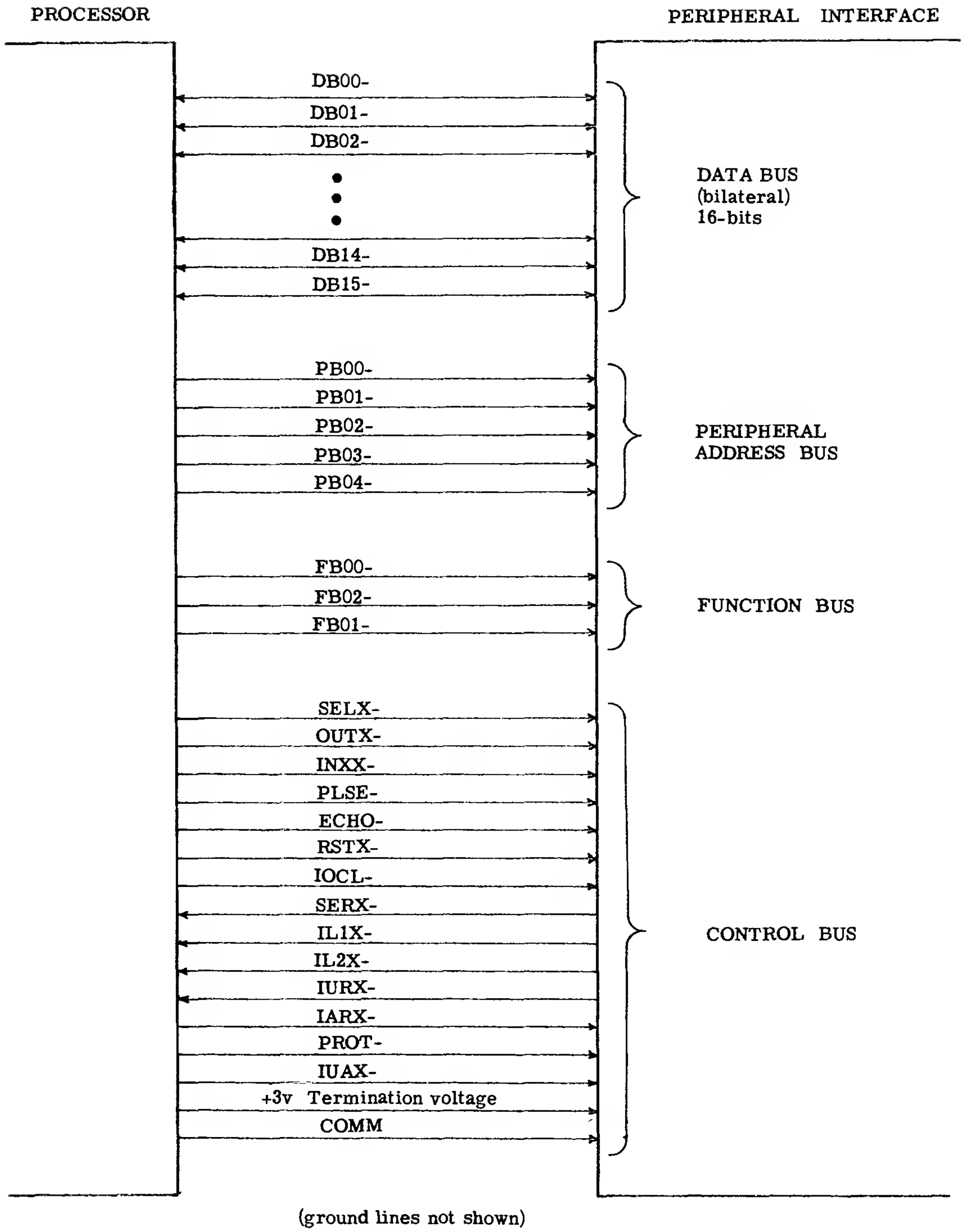


Figure 4-9 I/O Cable Buses

V INSTALLATION

PHYSICAL MOUNTING

The 816 chassis is designed to mount in a standard 19 inch rack or cabinet. It is 8 3/4 inches high and extends 17 inches behind the front rails. The chassis proper extends 16 inches and one inch is allowed for clearance of connectors on the rear of option I/O controller cards.

The power supply is 13 inches wide, 5 1/4 inch high and 5 1/4 inch deep. The power supply is mounted onto a bracket that is fastened to the back rails of a cabinet by four captive retractable panel screws. See Figure 5-1.

The 816 chassis is open at the top and bottom. Free vertical air flow will provide sufficient cooling for the system. When mounted in a cabinet with other equipment the use of a circulating air fan within the equipment cabinet is recommended.

POWER

The 816 needs only 250 watts of 60 cycle 115 volt power. The specifications on primary power are:

Voltage:	105 to 130 VRMS
Frequency:	50 Hz to 400 Hz

OPERATING ENVIRONMENT

Temperature:	0°c to 45°c
Humidity:	10% to 90% relative

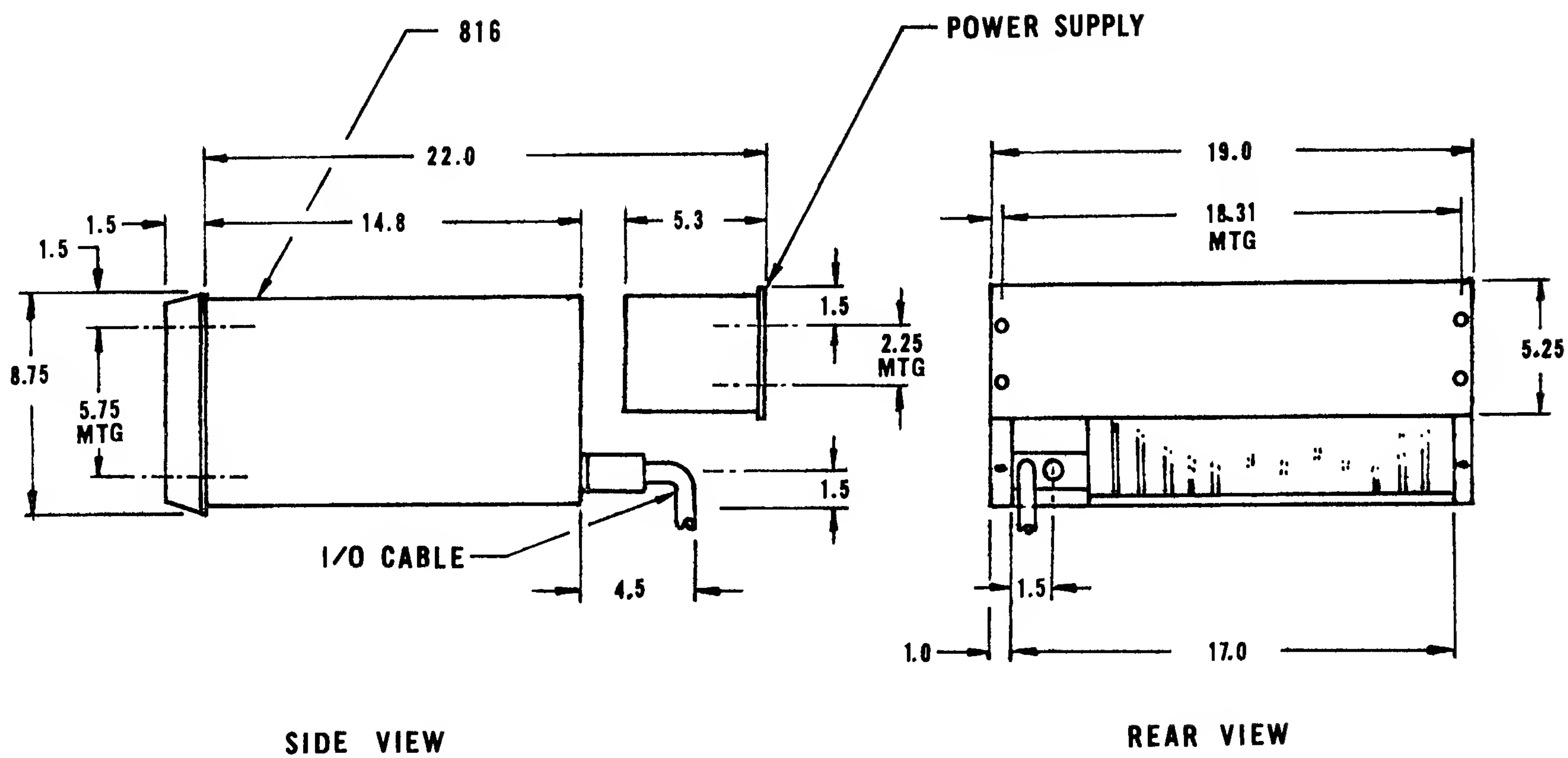


Figure 5-1 Installation Dimensions

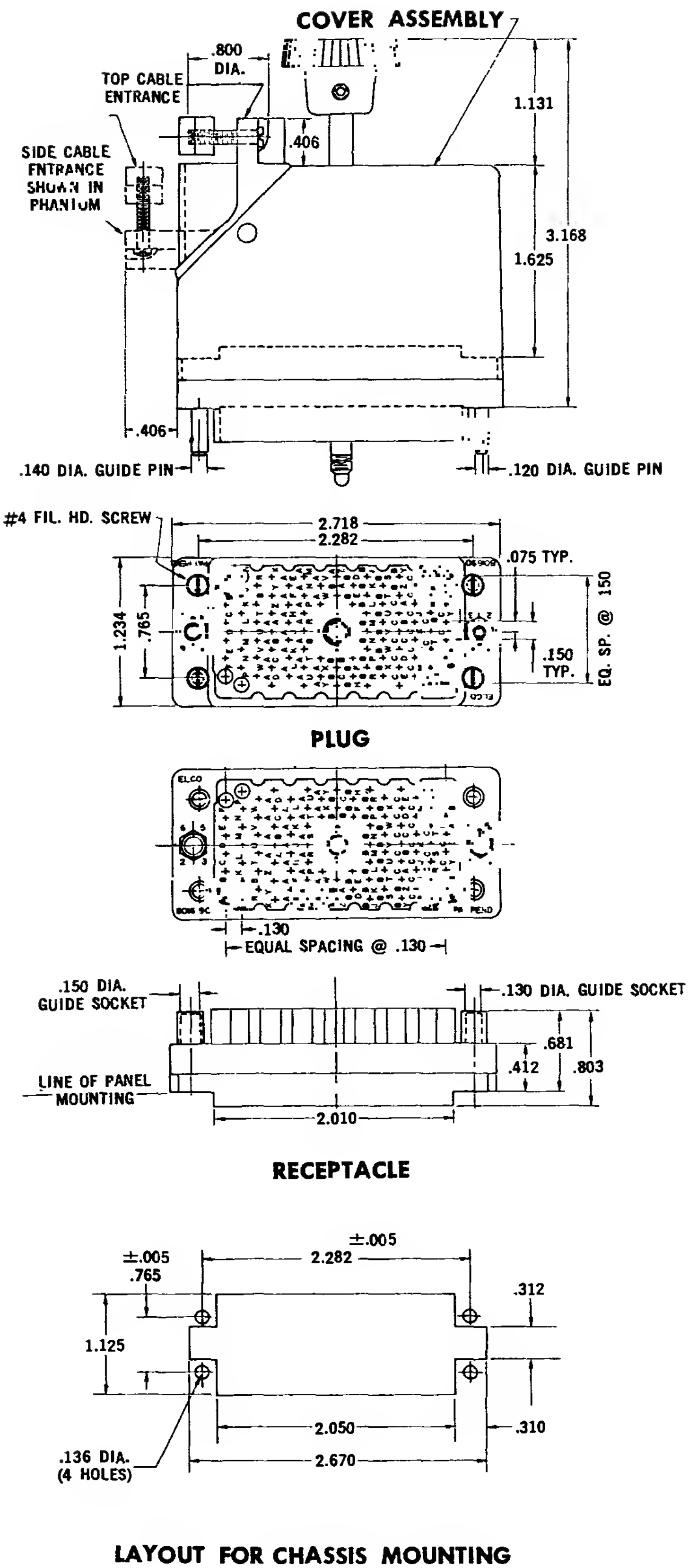


Figure 5-2 ELCO 90-pin Connector Dimensions

I/O CABLE TERMINATION LIST

Signal Name	Signal Pin	Return * Pin	Signal Name	Signal Pin	Return * Pin
DB00-	A	J	SERX-	BH	BC
DB01-	B	K	IL1X-	BS	BK
DB02-	C	L	IL2X-	BT	BL
DB03-	D	M	IURX-	BU	BM
DB04-	E	N	IARX-	BV	BN
DB05-	F	P	PROT-	BW	BP
DB06-	H	X	IUAX-	BX	BR
DB07-	R	Y	DB08-	CN	BY
PB00-	S	Z	DB09-	CF	BZ
PB01-	T	AA	DB10-	CH	CA
PB02-	U	AB	DB11-	CJ	CB
PB03-	V	AC	DB12-	CK	CC
PB04-	W	AD	DB13-	CL	CD
FB00-	AE	AN	DB14-	CM	CE
FB01-	AF	AP	DB15-	CW	CP
FB02-	AH	AR	Spare	CX	CR
EXCX-	AJ	AS	Spare	CY	CS
OUTX-	AK	AT	Spare	CZ	CT
INXX-	AL	AU	Spare	DA	CU
PLSE-	AV	AW	Spare	DB	CV
ECHO-	AZ	BA	COMM	AY	--
CLRX-	BD	BE	+3v	AM	--
IOCL-	BF	BB	Filter	AX	--

* Signal returns are connected to COMM

Connector type: ELCO Series 8016 VARICON/VARILOK

ORDERING INFORMATION:

Part # 00 - 8016 - 090 - XXX - XXX

Type of contact

Type of shell

CONTACT

000 - Crimp

296 - Wire Wrap

217 - Solder

SHELL

703 - Plug

707 - Receptacle

HEXADECIMAL ARITHMETIC

ADDITION TABLE

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
2	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
3	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
4	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
5	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14
6	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15
7	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16
8	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17
9	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18
A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19
B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A
C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

MULTIPLICATION TABLE

1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	04	06	08	0A	0C	0E	10	12	14	16	18	1A	1C	1E
3	06	09	0C	0F	12	15	18	1B	1E	21	24	27	2A	2D
4	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0A	0F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	1E	2B	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

HEXADECIMAL-DECIMAL INTEGER CONVERSION TABLE

The table below provides for direct conversions between hexadecimal integers in the range 0–FFF and decimal integers in the range 0–4095. For conversion of larger integers, the table values may be added to the following figures:

Hexadecimal	Decimal	Hexadecimal	Decimal
01 000	4 096	20 000	131 072
02 000	8 192	30 000	196 608
03 000	12 288	40 000	262 144
04 000	16 384	50 000	327 680
05 000	20 480	60 000	393 216
06 000	24 576	70 000	458 752
07 000	28 672	80 000	524 288
08 000	32 768	90 000	589 824
09 000	36 864	A0 000	655 360
0A 000	40 960	B0 000	720 896
0B 000	45 056	C0 000	786 432
0C 000	49 152	D0 000	851 968
0D 000	53 248	E0 000	917 504
0E 000	57 344	F0 000	983 040
0F 000	61 440	100 000	1 048 576
10 000	65 536	200 000	2 097 152
11 000	69 632	300 000	3 145 728
12 000	73 728	400 000	4 194 304
13 000	77 824	500 000	5 242 880
14 000	81 920	600 000	6 291 456
15 000	86 016	700 000	7 340 032
16 000	90 112	800 000	8 388 608
17 000	94 208	900 000	9 437 184
18 000	98 304	A00 000	10 485 760
19 000	102 400	B00 000	11 534 336
1A 000	106 496	C00 000	12 582 912
1B 000	110 592	D00 000	13 631 488
1C 000	114 688	E00 000	14 680 064
1D 000	118 784	F00 000	15 728 640
1E 000	122 880	1 000 000	16 777 216
1F 000	126 976	2 000 000	33 554 432

Hexadecimal fractions may be converted to decimal fractions as follows:

1. Express the hexadecimal fraction as an integer times 16^{-n}, where n is the number of significant hexadecimal places to the right of the hexadecimal point.

$$0.\text{CA9BF3}_{16} = \text{CA9 BF3}_{16} \times 16^{-6}$$

2. Find the decimal equivalent of the hexadecimal integer

$$\text{CA9 BF3}_{16} = 13\ 278\ 195_{10}$$

3. Multiply the decimal equivalent by 16^{-n}

$$\begin{array}{r} 13\ 278\ 195 \\ \underline{\times\ 596\ 046\ 448 \times 10^{-16}} \\ 0.791\ 442\ 096_{10} \end{array}$$

Decimal fractions may be converted to hexadecimal fractions by successively multiplying the decimal fraction by 16_{10}. After each multiplication, the integer portion is removed to form a hexadecimal fraction by building to the right of the hexadecimal point. However, since decimal arithmetic is used in this conversion, the integer portion of each product must be converted to hexadecimal numbers.

Example: Convert 0.895_{10} to its hexadecimal equivalent

$$\begin{array}{r} 0.895 \\ \underline{16} \\ 14.320 \\ \underline{16} \\ 5.120 \\ \underline{16} \\ 1.920 \\ \underline{16} \\ 14.720 \end{array} \qquad 0.\text{E51 E}_{16}$$

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
010	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
020	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
030	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
040	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
050	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
060	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
070	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
080	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
090	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A0	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B0	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C0	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D0	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E0	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F0	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
100	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
110	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
120	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
130	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
140	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
150	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
160	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
170	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
180	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
190	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A0	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B0	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C0	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D0	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E0	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F0	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511
200	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
210	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
220	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
230	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
240	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
250	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
260	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
270	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
280	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
290	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A0	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B0	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C0	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D0	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E0	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F0	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
300	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
310	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
320	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
330	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
340	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
350	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
360	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
370	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
380	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
390	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A0	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B0	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C0	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D0	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E0	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F0	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
400	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
410	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
420	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
430	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
440	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
450	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
460	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
470	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
480	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
490	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A0	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B0	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C0	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D0	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E0	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F0	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
500	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
510	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
520	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
530	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
540	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
550	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
560	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
570	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
580	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
590	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A0	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B0	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C0	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D0	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E0	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F0	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
600	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
610	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
620	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
630	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
640	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
650	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
660	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
670	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
680	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
690	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A0	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B0	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C0	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D0	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E0	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F0	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
700	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
710	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
720	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
730	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
740	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
750	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
760	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
770	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
780	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
790	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A0	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B0	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C0	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D0	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E0	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F0	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
800	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
810	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
820	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
830	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
840	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
860	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
870	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
880	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
890	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B0	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C0	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D0	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E0	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
900	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
910	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
920	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
930	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
950	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
960	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
970	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
980	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
990	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A0	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B0	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C0	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D0	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F0	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A00	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A10	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A20	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A40	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A50	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A60	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A70	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A90	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA0	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB0	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC0	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE0	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF0	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B00	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B10	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B30	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B40	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B50	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B60	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B70	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B80	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B90	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA0	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB0	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD0	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE0	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF0	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C10	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C20	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C30	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C40	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C60	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C70	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C80	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C90	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC0	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD0	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE0	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
D00	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D10	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D20	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D30	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D50	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D60	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D70	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D80	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB0	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC0	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD0	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF0	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583
E00	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E10	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E20	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E40	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E50	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E60	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E70	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E90	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA0	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB0	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC0	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE0	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF0	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F00	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F10	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F30	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F40	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F50	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F60	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F80	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F90	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA0	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB0	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD0	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE0	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF0	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

HEXADECIMAL-DECIMAL FRACTION CONVERSION TABLE

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.40 00 00 00	.25000 00000	.80 00 00 00	.50000 00000	.C0 00 00 00	.75000 00000
.01 00 00 00	.00390 62500	.41 00 00 00	.25390 62500	.81 00 00 00	.50390 62500	.C1 00 00 00	.75390 62500
.02 00 00 00	.00781 25000	.42 00 00 00	.25781 25000	.82 00 00 00	.50781 25000	.C2 00 00 00	.75781 25000
.03 00 00 00	.01171 87500	.43 00 00 00	.26171 87500	.83 00 00 00	.51171 87500	.C3 00 00 00	.76171 87500
.04 00 00 00	.01562 50000	.44 00 00 00	.26562 50000	.84 00 00 00	.51562 50000	.C4 00 00 00	.76562 50000
.05 00 00 00	.01953 12500	.45 00 00 00	.26953 12500	.85 00 00 00	.51953 12500	.C5 00 00 00	.76953 12500
.06 00 00 00	.02343 75000	.46 00 00 00	.27343 75000	.86 00 00 00	.52343 75000	.C6 00 00 00	.77343 75000
.07 00 00 00	.02734 37500	.47 00 00 00	.27734 37500	.87 00 00 00	.52734 37500	.C7 00 00 00	.77734 37500
.08 00 00 00	.03125 00000	.48 00 00 00	.28125 00000	.88 00 00 00	.53125 00000	.C8 00 00 00	.78125 00000
.09 00 00 00	.03515 62500	.49 00 00 00	.28515 62500	.89 00 00 00	.53515 62500	.C9 00 00 00	.78515 62500
.0A 00 00 00	.03906 25000	.4A 00 00 00	.28906 25000	.8A 00 00 00	.53906 25000	.CA 00 00 00	.78906 25000
.0B 00 00 00	.04296 87500	.4B 00 00 00	.29296 87500	.8B 00 00 00	.54296 87500	.CB 00 00 00	.79296 87500
.0C 00 00 00	.04687 50000	.4C 00 00 00	.29687 50000	.8C 00 00 00	.54687 50000	.CC 00 00 00	.79687 50000
.0D 00 00 00	.05078 12500	.4D 00 00 00	.30078 12500	.8D 00 00 00	.55078 12500	.CD 00 00 00	.80078 12500
.0E 00 00 00	.05468 75000	.4E 00 00 00	.30468 75000	.8E 00 00 00	.55468 75000	.CE 00 00 00	.80468 75000
.0F 00 00 00	.05859 37500	.4F 00 00 00	.30859 37500	.8F 00 00 00	.55859 37500	.CF 00 00 00	.80859 37500
.10 00 00 00	.06250 00000	.50 00 00 00	.31250 00000	.90 00 00 00	.56250 00000	.D0 00 00 00	.81250 00000
.11 00 00 00	.06640 62500	.51 00 00 00	.31640 62500	.91 00 00 00	.56640 62500	.D1 00 00 00	.81640 62500
.12 00 00 00	.07031 25000	.52 00 00 00	.32031 25000	.92 00 00 00	.57031 25000	.D2 00 00 00	.82031 25000
.13 00 00 00	.07421 87500	.53 00 00 00	.32421 87500	.93 00 00 00	.57421 87500	.D3 00 00 00	.82421 87500
.14 00 00 00	.07812 50000	.54 00 00 00	.32812 50000	.94 00 00 00	.57812 50000	.D4 00 00 00	.82812 50000
.15 00 00 00	.08203 12500	.55 00 00 00	.33203 12500	.95 00 00 00	.58203 12500	.D5 00 00 00	.83203 12500
.16 00 00 00	.08593 75000	.56 00 00 00	.33593 75000	.96 00 00 00	.58593 75000	.D6 00 00 00	.83593 75000
.17 00 00 00	.08984 37500	.57 00 00 00	.33984 37500	.97 00 00 00	.58984 37500	.D7 00 00 00	.83984 37500
.18 00 00 00	.09375 00000	.58 00 00 00	.34375 00000	.98 00 00 00	.59375 00000	.D8 00 00 00	.84375 00000
.19 00 00 00	.09765 62500	.59 00 00 00	.34765 62500	.99 00 00 00	.59765 62500	.D9 00 00 00	.84765 62500
.1A 00 00 00	.10156 25000	.5A 00 00 00	.35156 25000	.9A 00 00 00	.60156 25000	.DA 00 00 00	.85156 25000
.1B 00 00 00	.10546 87500	.5B 00 00 00	.35546 87500	.9B 00 00 00	.60546 87500	.DB 00 00 00	.85546 87500
.1C 00 00 00	.10937 50000	.5C 00 00 00	.35937 50000	.9C 00 00 00	.60937 50000	.DC 00 00 00	.85937 50000
.1D 00 00 00	.11328 12500	.5D 00 00 00	.36328 12500	.9D 00 00 00	.61328 12500	.DD 00 00 00	.86328 12500
.1E 00 00 00	.11718 75000	.5E 00 00 00	.36718 75000	.9E 00 00 00	.61718 75000	.DE 00 00 00	.86718 75000
.1F 00 00 00	.12109 37500	.5F 00 00 00	.37109 37500	.9F 00 00 00	.62109 37500	.DF 00 00 00	.87109 37500
.20 00 00 00	.12500 00000	.60 00 00 00	.37500 00000	.A0 00 00 00	.62500 00000	.E0 00 00 00	.87500 00000
.21 00 00 00	.12890 62500	.61 00 00 00	.37890 62500	.A1 00 00 00	.62890 62500	.E1 00 00 00	.87890 62500
.22 00 00 00	.13281 25000	.62 00 00 00	.38281 25000	.A2 00 00 00	.63281 25000	.E2 00 00 00	.88281 25000
.23 00 00 00	.13671 87500	.63 00 00 00	.38671 87500	.A3 00 00 00	.63671 87500	.E3 00 00 00	.88671 87500
24 00 00 00	.14062 50000	.64 00 00 00	.39062 50000	.A4 00 00 00	.64062 50000	.E4 00 00 00	.89062 50000
.25 00 00 00	.14453 12500	.65 00 00 00	.39453 12500	.A5 00 00 00	.64453 12500	.E5 00 00 00	.89453 12500
.26 00 00 00	.14843 75000	.66 00 00 00	.39843 75000	.A6 00 00 00	.64843 75000	.E6 00 00 00	.89843 75000
.27 00 00 00	.15234 37500	.67 00 00 00	.40234 37500	.A7 00 00 00	.65234 37500	.E7 00 00 00	.90234 37500
.28 00 00 00	.15625 00000	.68 00 00 00	.40625 00000	.A8 00 00 00	.65625 00000	.E8 00 00 00	.90625 00000
.29 00 00 00	.16015 62500	.69 00 00 00	.41015 62500	.A9 00 00 00	.66015 62500	.E9 00 00 00	.91015 62500
.2A 00 00 00	.16406 25000	.6A 00 00 00	.41406 25000	.AA 00 00 00	.66406 25000	.EA 00 00 00	.91406 25000
.2B 00 00 00	.16796 87500	.6B 00 00 00	.41796 87500	.AB 00 00 00	.66796 87500	.EB 00 00 00	.91796 B7500
.2C 00 00 00	.17187 50000	.6C 00 00 00	.42187 50000	.AC 00 00 00	.67187 50000	.EC 00 00 00	.92187 50000
.2D 00 00 00	.17578 12500	.6D 00 00 00	.42578 12500	.AD 00 00 00	.67578 12500	.ED 00 00 00	.92578 12500
.2E 00 00 00	.17968 75000	.6E 00 00 00	.42968 75000	.AE 00 00 00	.67968 75000	.EE 00 00 00	.92968 75000
.2F 00 00 00	.18359 37500	.6F 00 00 00	.43359 37500	.AF 00 00 00	.68359 37500	.EF 00 00 00	.93359 37500
30 00 00 00	.18750 00000	.70 00 00 00	.43750 00000	.B0 00 00 00	.68750 00000	.F0 00 00 00	.93750 00000
.31 00 00 00	.19140 62500	.71 00 00 00	.44140 62500	.B1 00 00 00	.69140 62500	.F1 00 00 00	.94140 62500
.32 00 00 00	.19531 25000	.72 00 00 00	.44531 25000	.B2 00 00 00	.69531 25000	.F2 00 00 00	.94531 25000
.33 00 00 00	.19921 87500	.73 00 00 00	.44921 87500	.B3 00 00 00	.69921 87500	.F3 00 00 00	.94921 87500
.34 00 00 00	.20312 50000	.74 00 00 00	.45312 50000	.B4 00 00 00	.70312 50000	.F4 00 00 00	.95312 50000
35 00 00 00	.20703 12500	.75 00 00 00	.45703 12500	.B5 00 00 00	.70703 12500	.F5 00 00 00	.95703 12500
.36 00 00 00	.21093 75000	.76 00 00 00	.46093 75000	.86 00 00 00	.71093 75000	.F6 00 00 00	.96093 75000
.37 00 00 00	.21484 37500	.77 00 00 00	.46484 37500	.B7 00 00 00	.71484 37500	.F7 00 00 00	.96484 37500
38 00 00 00	.21875 00000	.78 00 00 00	.46875 00000	.B8 00 00 00	.71875 00000	.F8 00 00 00	.96875 00000
.39 00 00 00	.22265 62500	.79 00 00 00	.47265 62500	.B9 00 00 00	.72265 62500	.F9 00 00 00	.97265 62500
.3A 00 00 00	.22656 25000	.7A 00 00 00	.47656 25000	.BA 00 00 00	.72656 25000	.FA 00 00 00	.97656 25000
3B 00 00 00	.23046 87500	.7B 00 00 00	.48046 87500	.BB 00 00 00	.73046 87500	.FB 00 00 00	.98046 87500
.3C 00 00 00	.23437 50000	.7C 00 00 00	.48437 50000	.BC 00 00 00	.73437 50000	.FC 00 00 00	.98437 50000
3D 00 00 00	.23828 12500	.7D 00 00 00	.48828 12500	.BD 00 00 00	.73828 12500	.FD 00 00 00	.98828 12500
3E 00 00 00	.24218 75000	.7E 00 00 00	.49218 75000	.BE 00 00 00	.74218 75000	.FE 00 00 00	.99218 75000
.3F 00 00 00	24609 37500	.7F 00 00 00	.49609 37500	.BF 00 00 00	.74609 37500	.FF 00 00 00	.99609 37500

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 40 00 00	.00097 65625	.00 80 00 00	.00195 31250	.00 C0 00 00	.00292 96875
.00 01 00 00	.00001 52587	.00 41 00 00	.00099 18212	.00 81 00 00	.00196 83837	.00 C1 00 00	.00294 49462
.00 02 00 00	.00003 05175	.00 42 00 00	.00100 70800	.00 82 00 00	.00198 36425	.00 C2 00 00	.00296 02050
.00 03 00 00	.00004 57763	.00 43 00 00	.00102 23388	.00 83 00 00	.00199 89013	.00 C3 00 00	.00297 54638
.00 04 00 00	.00006 10351	.00 44 00 00	.00103 75976	.00 84 00 00	.00201 41601	.00 C4 00 00	.00299 07226
.00 05 00 00	.00007 62939	.00 45 00 00	.00105 28564	.00 85 00 00	.00202 94189	.00 C5 00 00	.00300 59814
.00 06 00 00	.00009 15527	.00 46 00 00	.00106 81152	.00 86 00 00	.00204 46777	.00 C6 00 00	.00302 12402
.00 07 00 00	.00010 68115	.00 47 00 00	.00108 33740	.00 87 00 00	.00205 99365	.00 C7 00 00	.00303 64990
.00 08 00 00	.00012 20703	.00 48 00 00	.00109 86328	.00 88 00 00	.00207 51953	.00 C8 00 00	.00305 17578
.00 09 00 00	.00013 73291	.00 49 00 00	.00111 38916	.00 89 00 00	.00209 04541	.00 C9 00 00	.00306 70166
.00 0A 00 00	.00015 25878	.00 4A 00 00	.00112 91503	.00 8A 00 00	.00210 57128	.00 CA 00 00	.00308 22753
.00 0B 00 00	.00016 78466	.00 4B 00 00	.00114 44091	.00 8B 00 00	.00212 09716	.00 CB 00 00	.00309 75341
.00 0C 00 00	.00018 31054	.00 4C 00 00	.00115 96679	.00 8C 00 00	.00213 62304	.00 CC 00 00	.00311 27929
.00 0D 00 00	.00019 83642	.00 4D 00 00	.00117 49267	.00 8D 00 00	.00215 14892	.00 CD 00 00	.00312 80517
.00 0E 00 00	.00021 36230	.00 4E 00 00	.00119 01855	.00 8E 00 00	.00216 67480	.00 CE 00 00	.00314 33105
.00 0F 00 00	.00022 88818	.00 4F 00 00	.00120 54443	.00 8F 00 00	.00218 20068	.00 CF 00 00	.00315 85693
.00 10 00 00	.00024 41406	.00 50 00 00	.00122 07031	.00 90 00 00	.00219 72656	.00 D0 00 00	.00317 38281
.00 11 00 00	.00025 93994	.00 51 00 00	.00123 59619	.00 91 00 00	.00221 25244	.00 D1 00 00	.00318 90869
.00 12 00 00	.00027 46582	.00 52 00 00	.00125 12207	.00 92 00 00	.00222 77832	.00 D2 00 00	.00320 43457
.00 13 00 00	.00028 99169	.00 53 00 00	.00126 64794	.00 93 00 00	.00224 30419	.00 D3 00 00	.00321 96044
.00 14 00 00	.00030 51757	.00 54 00 00	.00128 17382	.00 94 00 00	.00225 83007	.00 D4 00 00	.00323 48632
.00 15 00 00	.00032 04345	.00 55 00 00	.00129 69970	.00 95 00 00	.00227 35595	.00 D5 00 00	.00325 01220
.00 16 00 00	.00033 56933	.00 56 00 00	.00131 22558	.00 96 00 00	.00228 88183	.00 D6 00 00	.00326 53808
.00 17 00 00	.00035 09521	.00 57 00 00	.00132 75146	.00 97 00 00	.00230 40771	.00 D7 00 00	.00328 06396
.00 18 00 00	.00036 62109	.00 58 00 00	.00134 27734	.00 98 00 00	.00231 93359	.00 D8 00 00	.00329 58984
.00 19 00 00	.00038 14697	.00 59 00 00	.00135 80322	.00 99 00 00	.00233 45947	.00 D9 00 00	.00331 11572
.00 1A 00 00	.00039 67285	.00 5A 00 00	.00137 32910	.00 9A 00 00	.00234 98535	.00 DA 00 00	.00332 64160
.00 1B 00 00	.00041 19873	.00 5B 00 00	.00138 85498	.00 9B 00 00	.00236 51123	.00 DB 00 00	.00334 16748
.00 1C 00 00	.00042 72460	.00 5C 00 00	.00140 38085	.00 9C 00 00	.00238 03710	.00 DC 00 00	.00335 69335
.00 1D 00 00	.00044 25048	.00 5D 00 00	.00141 90673	.00 9D 00 00	.00239 56298	.00 DD 00 00	.00337 21923
.00 1E 00 00	.00045 77636	.00 5E 00 00	.00143 43261	.00 9E 00 00	.00241 08886	.00 DE 00 00	.00338 74511
.00 1F 00 00	.00047 30224	.00 5F 00 00	.00144 95849	.00 9F 00 00	.00242 61474	.00 DF 00 00	.00340 27099
.00 20 00 00	.00048 82812	.00 60 00 00	.00146 48437	.00 A0 00 00	.00244 14062	.00 E0 00 00	.00341 79687
.00 21 00 00	.00050 35400	.00 61 00 00	.00148 01025	.00 A1 00 00	.00245 66650	.00 E1 00 00	.00343 32275
.00 22 00 00	.00051 87988	.00 62 00 00	.00149 53613	.00 A2 00 00	.00247 19238	.00 E2 00 00	.00344 84863
.00 23 00 00	.00053 40576	.00 63 00 00	.00151 06201	.00 A3 00 00	.00248 71826	.00 E3 00 00	.00346 37451
.00 24 00 00	.00054 93164	.00 64 00 00	.00152 58789	.00 A4 00 00	.00250 24414	.00 E4 00 00	.00347 90039
.00 25 00 00	.00056 45751	.00 65 00 00	.00154 11376	.00 A5 00 00	.00251 77001	.00 E5 00 00	.00349 42626
.00 26 00 00	.00057 98339	.00 66 00 00	.00155 63964	.00 A6 00 00	.00253 29589	.00 E6 00 00	.00350 95214
.00 27 00 00	.00059 50927	.00 67 00 00	.00157 16552	.00 A7 00 00	.00254 82177	.00 E7 00 00	.00352 47802
.00 28 00 00	.00061 03515	.00 68 00 00	.00158 69140	.00 A8 00 00	.00256 34765	.00 E8 00 00	.00354 00390
.00 29 00 00	.00062 56103	.00 69 00 00	.00160 21728	.00 A9 00 00	.00257 87353	.00 E9 00 00	.00355 52978
.00 2A 00 00	.00064 08691	.00 6A 00 00	.00161 74316	.00 AA 00 00	.00259 39941	.00 EA 00 00	.00357 05566
.00 2B 00 00	.00065 61279	.00 6B 00 00	.00163 26904	.00 AB 00 00	.00260 92529	.00 EB 00 00	.00358 58154
.00 2C 00 00	.00067 13867	.00 6C 00 00	.00164 79492	.00 AC 00 00	.00262 45117	.00 EC 00 00	.00360 10742
.00 2D 00 00	.00068 66455	.00 6D 00 00	.00166 32080	.00 AD 00 00	.00263 97705	.00 ED 00 00	.00361 63330
.00 2E 00 00	.00070 19042	.00 6E 00 00	.00167 84667	.00 AE 00 00	.00265 50292	.00 EE 00 00	.00363 15917
.00 2F 00 00	.00071 71630	.00 6F 00 00	.00169 37255	.00 AF 00 00	.00267 02880	.00 EF 00 00	.00364 68505
.00 30 00 00	.00073 24218	.00 70 00 00	.00170 89843	.00 B0 00 00	.00268 55468	.00 F0 00 00	.00366 21093
.00 31 00 00	.00074 76806	.00 71 00 00	.00172 42431	.00 B1 00 00	.00270 08056	.00 F1 00 00	.00367 73681
.00 32 00 00	.00076 29394	.00 72 00 00	.00173 95019	.00 B2 00 00	.00271 60644	.00 F2 00 00	.00369 26269
.00 33 00 00	.00077 81982	.00 73 00 00	.00175 47607	.00 B3 00 00	.00273 13232	.00 F3 00 00	.00370 78857
.00 34 00 00	.00079 34570	.00 74 00 00	.00177 00195	.00 B4 00 00	.00274 65820	.00 F4 00 00	.00372 31445
.00 35 00 00	.00080 87158	.00 75 00 00	.00178 52783	.00 B5 00 00	.00276 18408	.00 F5 00 00	.00373 84033
.00 36 00 00	.00082 39746	.00 76 00 00	.00180 05371	.00 B6 00 00	.00277 70996	.00 F6 00 00	.00375 36621
.00 37 00 00	.00083 92333	.00 77 00 00	.00181 57958	.00 B7 00 00	.00279 23583	.00 F7 00 00	.00376 89208
.00 38 00 00	.00085 44921	.00 78 00 00	.00183 10546	.00 B8 00 00	.00280 76171	.00 F8 00 00	.00378 41796
.00 39 00 00	.00086 97509	.00 79 00 00	.00184 63134	.00 B9 00 00	.00282 28759	.00 F9 00 00	.00379 94384
.00 3A 00 00	.00088 50097	.00 7A 00 00	.00186 15722	.00 BA 00 00	.00283 81347	.00 FA 00 00	.00381 46972
.00 3B 00 00	.00090 02685	.00 7B 00 00	.00187 68310	.00 BB 00 00	.00285 33935	.00 FB 00 00	.00382 99560
.00 3C 00 00	.00091 55273	.00 7C 00 00	.00189 20898	.00 BC 00 00	.00286 86523	.00 FC 00 00	.00384 52148
.00 3D 00 00	.00093 07861	.00 7D 00 00	.00190 73486	.00 BD 00 00	.00288 39111	.00 FD 00 00	.00386 04736
.00 3E 00 00	.00094 60449	.00 7E 00 00	.00192 26074	.00 BE 00 00	.00289 91699	.00 FE 00 00	.00387 57324
.00 3F 00 00	.00096 13037	.00 7F 00 00	.00193 78662	.00 BF 00 00	.00291 44287	.00 FF 00 00	.00389 09912

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 00 40 00	.00000 38146	.00 00 80 00	.00000 76293	.00 00 C0 00	.00001 14440
.00 00 01 00	.00000 00596	.00 00 41 00	.00000 38743	.00 00 81 00	.00000 76889	.00 00 C1 00	.00001 15036
.00 00 02 00	.00000 01192	.00 00 42 00	.00000 39339	.00 00 82 00	.00000 77486	.00 00 C2 00	.00001 15633
.00 00 03 00	.00000 01788	.00 00 43 00	.00000 39935	.00 00 83 00	.00000 78082	.00 00 C3 00	.00001 16229
.00 00 04 00	.00000 02384	.00 00 44 00	.00000 40531	.00 00 84 00	.00000 78678	.00 00 C4 00	.00001 16825
.00 00 05 00	.00000 02980	.00 00 45 00	.00000 41127	.00 00 85 00	.00000 79274	.00 00 C5 00	.00001 17421
.00 00 06 00	.00000 03576	.00 00 46 00	.00000 41723	.00 00 86 00	.00000 79870	.00 00 C6 00	.00001 18017
.00 00 07 00	.00000 04172	.00 00 47 00	.00000 42319	.00 00 87 00	.00000 80466	.00 00 C7 00	.00001 18613
.00 00 08 00	.00000 04768	.00 00 48 00	.00000 42915	.00 00 88 00	.00000 81062	.00 00 C8 00	.00001 19209
.00 00 09 00	.00000 05364	.00 00 49 00	.00000 43511	.00 00 89 00	.00000 81658	.00 00 C9 00	.00001 19805
.00 00 0A 00	.00000 05960	.00 00 4A 00	.00000 44107	.00 00 8A 00	.00000 82254	.00 00 CA 00	.00001 20401
.00 00 0B 00	.00000 06556	.00 00 4B 00	.00000 44703	.00 00 8B 00	.00000 82850	.00 00 CB 00	.00001 20997
.00 00 0C 00	.00000 07152	.00 00 4C 00	.00000 45299	.00 00 8C 00	.00000 83446	.00 00 CC 00	.00001 21593
.00 00 0D 00	.00000 07748	.00 00 4D 00	.00000 45895	.00 00 8D 00	.00000 84042	.00 00 CD 00	.00001 22189
.00 00 0E 00	.00000 08344	.00 00 4E 00	.00000 46491	.00 00 8E 00	.00000 84638	.00 00 CE 00	.00001 22785
.00 00 0F 00	.00000 08940	.00 00 4F 00	.00000 47087	.00 00 8F 00	.00000 85234	.00 00 CF 00	.00001 23381
.00 00 10 00	.00000 09536	.00 00 50 00	.00000 47683	.00 00 90 00	.00000 85830	.00 00 D0 00	.00001 23977
.00 00 11 00	.00000 10132	.00 00 51 00	.00000 48279	.00 00 91 00	.00000 86426	.00 00 D1 00	.00001 24573
.00 00 12 00	.00000 10728	.00 00 52 00	.00000 48875	.00 00 92 00	.00000 87022	.00 00 D2 00	.00001 25169
.00 00 13 00	.00000 11324	.00 00 53 00	.00000 49471	.00 00 93 00	.00000 87618	.00 00 D3 00	.00001 25765
.00 00 14 00	.00000 11920	.00 00 54 00	.00000 50067	.00 00 94 00	.00000 88214	.00 00 D4 00	.00001 26361
.00 00 15 00	.00000 12516	.00 00 55 00	.00000 50663	.00 00 95 00	.00000 88810	.00 00 D5 00	.00001 26957
.00 00 16 00	.00000 13113	.00 00 56 00	.00000 51259	.00 00 96 00	.00000 89406	.00 00 D6 00	.00001 27553
.00 00 17 00	.00000 13709	.00 00 57 00	.00000 51856	.00 00 97 00	.00000 90003	.00 00 D7 00	.00001 28149
.00 00 18 00	.00000 14305	.00 00 58 00	.00000 52452	.00 00 98 00	.00000 90599	.00 00 D8 00	.00001 28746
.00 00 19 00	.00000 14901	.00 00 59 00	.00000 53048	.00 00 99 00	.00000 91195	.00 00 D9 00	.00001 29342
.00 00 1A 00	.00000 15497	.00 00 5A 00	.00000 53644	.00 00 9A 00	.00000 91791	.00 00 DA 00	.00001 29938
.00 00 1B 00	.00000 16093	.00 00 5B 00	.00000 54240	.00 00 9B 00	.00000 92387	.00 00 DB 00	.00001 30534
.00 00 1C 00	.00000 16689	.00 00 5C 00	.00000 54836	.00 00 9C 00	.00000 92983	.00 00 DC 00	.00001 31130
.00 00 1D 00	.00000 17285	.00 00 5D 00	.00000 55432	.00 00 9D 00	.00000 93579	.00 00 DD 00	.00001 31726
.00 00 1E 00	.00000 17881	.00 00 5E 00	.00000 56028	.00 00 9E 00	.00000 94175	.00 00 DE 00	.00001 32322
.00 00 1F 00	.00000 18477	.00 00 5F 00	.00000 56624	.00 00 9F 00	.00000 94771	.00 00 DF 00	.00001 32918
.00 00 20 00	.00000 19073	.00 00 60 00	.00000 57220	.00 00 A0 00	.00000 95367	.00 00 E0 00	.00001 33514
.00 00 21 00	.00000 19669	.00 00 61 00	.00000 57816	.00 00 A1 00	.00000 95963	.00 00 E1 00	.00001 34110
.00 00 22 00	.00000 20265	.00 00 62 00	.00000 58412	.00 00 A2 00	.00000 96559	.00 00 E2 00	.00001 34706
.00 00 23 00	.00000 20861	.00 00 63 00	.00000 59008	.00 00 A3 00	.00000 97155	.00 00 E3 00	.00001 35302
.00 00 24 00	.00000 21457	.00 00 64 00	.00000 59604	.00 00 A4 00	.00000 97751	.00 00 E4 00	.00001 35898
.00 00 25 00	.00000 22053	.00 00 65 00	.00000 60200	.00 00 A5 00	.00000 98347	.00 00 E5 00	.00001 36494
.00 00 26 00	.00000 22649	.00 00 66 00	.00000 60796	.00 00 A6 00	.00000 98943	.00 00 E6 00	.00001 37090
.00 00 27 00	.00000 23245	.00 00 67 00	.00000 61392	.00 00 A7 00	.00000 99539	.00 00 E7 00	.00001 37686
.00 00 28 00	.00000 23841	.00 00 68 00	.00000 61988	.00 00 A8 00	.00001 00135	.00 00 E8 00	.00001 38282
.00 00 29 00	.00000 24437	.00 00 69 00	.00000 62584	.00 00 A9 00	.00001 00731	.00 00 E9 00	.00001 38878
.00 00 2A 00	.00000 25033	.00 00 6A 00	.00000 63180	.00 00 AA 00	.00001 01327	.00 00 EA 00	.00001 39474
.00 00 2B 00	.00000 25629	.00 00 6B 00	.00000 63776	.00 00 AB 00	.00001 01923	.00 00 EB 00	.00001 40070
.00 00 2C 00	.00000 26226	.00 00 6C 00	.00000 64373	.00 00 AC 00	.00001 02519	.00 00 EC 00	.00001 40666
.00 00 2D 00	.00000 26822	.00 00 6D 00	.00000 64969	.00 00 AD 00	.00001 03116	.00 00 ED 00	.00001 41263
.00 00 2E 00	.00000 27418	.00 00 6E 00	.00000 65565	.00 00 AE 00	.00001 03712	.00 00 EE 00	.00001 41859
.00 00 2F 00	.00000 28014	.00 00 6F 00	.00000 66161	.00 00 AF 00	.00001 04308	.00 00 EF 00	.00001 42455
.00 00 30 00	.00000 28610	.00 00 70 00	.00000 66757	.00 00 B0 00	.00001 04904	.00 00 F0 00	.00001 43051
.00 00 31 00	.00000 29206	.00 00 71 00	.00000 67353	.00 00 B1 00	.00001 05500	.00 00 F1 00	.00001 43647
.00 00 32 00	.00000 29802	.00 00 72 00	.00000 67949	.00 00 B2 00	.00001 06096	.00 00 F2 00	.00001 44243
.00 00 33 00	.00000 30398	.00 00 73 00	.00000 68545	.00 00 B3 00	.00001 06692	.00 00 F3 00	.00001 44839
.00 00 34 00	.00000 30994	.00 00 74 00	.00000 69141	.00 00 B4 00	.00001 07288	.00 00 F4 00	.00001 45435
.00 00 35 00	.00000 31590	.00 00 75 00	.00000 69737	.00 00 B5 00	.00001 07884	.00 00 F5 00	.00001 46031
.00 00 36 00	.00000 32186	.00 00 76 00	.00000 70333	.00 00 86 00	.00001 08480	.00 00 F6 00	.00001 46627
.00 00 37 00	.00000 32782	.00 00 77 00	.00000 70929	.00 00 B7 00	.00001 09076	.00 00 F7 00	.00001 47223
.00 00 38 00	.00000 33378	.00 00 78 00	.00000 71525	.00 00 B8 00	.00001 09672	.00 00 F8 00	.00001 47819
.00 00 39 00	.00000 33974	.00 00 79 00	.00000 72121	.00 00 B9 00	.00001 10268	.00 00 F9 00	.00001 48415
.00 00 3A 00	.00000 34570	.00 00 7A 00	.00000 72717	.00 00 BA 00	.00001 10864	.00 00 FA 00	.00001 49011
.00 00 3B 00	.00000 35166	.00 00 7B 00	.00000 73313	.00 00 BB 00	.00001 11460	.00 00 FB 00	.00001 49607
.00 00 3C 00	.00000 35762	.00 00 7C 00	.00000 73909	.00 00 BC 00	.00001 12056	.00 00 FC 00	.00001 50203
.00 00 3D 00	.00000 36358	.00 00 7D 00	.00000 74505	.00 00 BD 00	.00001 12652	.00 00 FD 00	.00001 50799
.00 00 3E 00	.00000 36954	.00 00 7E 00	.00000 75101	.00 00 BE 00	.00001 13248	.00 00 FE 00	.00001 51395
.00 00 3F 00	.00000 37550	.00 00 7F 00	.00000 75697	.00 00 BF 00	.00001 13844	.00 00 FF 00	.00001 51991

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 00 00 40	.00000 00149	.00 00 00 80	.00000 00298	.00 00 00 C0	.00000 00447
.00 00 00 01	.00000 00002	.00 00 00 41	.00000 00151	.00 00 00 81	.00000 00300	.00 00 00 C1	.00000 00449
.00 00 00 02	.00000 00004	.00 00 00 42	.00000 00153	.00 00 00 82	.00000 00302	.00 00 00 C2	.00000 00451
.00 00 00 03	.00000 00006	.00 00 00 43	.00000 00155	.00 00 00 83	.00000 00305	.00 00 00 C3	.00000 00454
.00 00 00 04	.00000 00009	.00 00 00 44	.00000 00158	.00 00 00 84	.00000 00307	.00 00 00 C4	.00000 00456
.00 00 00 05	.00000 00011	.00 00 00 45	.00000 00160	.00 00 00 85	.00000 00309	.00 00 00 C5	.00000 00458
.00 00 00 06	.00000 00013	.00 00 00 46	.00000 00162	.00 00 00 86	.00000 00311	.00 00 00 C6	.00000 00461
.00 00 00 07	.00000 00016	.00 00 00 47	.00000 00165	.00 00 00 87	.00000 00314	.00 00 00 C7	.00000 00463
.00 00 00 08	.00000 00018	.00 00 00 48	.00000 00167	.00 00 00 88	.00000 00316	.00 00 00 C8	.00000 00465
.00 00 00 09	.00000 00020	.00 00 00 49	.00000 00169	.00 00 00 89	.00000 00318	.00 00 00 C9	.00000 00467
.00 00 00 0A	.00000 00023	.00 00 00 4A	.00000 00172	.00 00 00 8A	.00000 00321	.00 00 00 CA	.00000 00470
.00 00 00 0B	.00000 00025	.00 00 00 4B	.00000 00174	.00 00 00 8B	.00000 00323	.00 00 00 CB	.00000 00472
.00 00 00 0C	.00000 00027	.00 00 00 4C	.00000 00176	.00 00 00 8C	.00000 00325	.00 00 00 CC	.00000 00474
.00 00 00 0D	.00000 00030	.00 00 00 4D	.00000 00179	.00 00 00 8D	.00000 00328	.00 00 00 CD	.00000 00477
.00 00 00 0E	.00000 00032	.00 00 00 4E	.00000 00181	.00 00 00 8E	.00000 00330	.00 00 00 CE	.00000 00479
.00 00 00 0F	.00000 00034	.00 00 00 4F	.00000 00183	.00 00 00 8F	.00000 00332	.00 00 00 CF	.00000 00481
.00 00 00 10	.00000 00037	.00 00 00 50	.00000 00186	.00 00 00 90	.00000 00335	.00 00 00 D0	.00000 00484
.00 00 00 11	.00000 00039	.00 00 00 51	.00000 00188	.00 00 00 91	.00000 00337	.00 00 00 D1	.00000 00486
.00 00 00 12	.00000 00041	.00 00 00 52	.00000 00190	.00 00 00 92	.00000 00339	.00 00 00 D2	.00000 00488
.00 00 00 13	.00000 00044	.00 00 00 53	.00000 00193	.00 00 00 93	.00000 00342	.00 00 00 D3	.00000 00491
.00 00 00 14	.00000 00046	.00 00 00 54	.00000 00195	.00 00 00 94	.00000 00344	.00 00 00 D4	.00000 00493
.00 00 00 15	.00000 00048	.00 00 00 55	.00000 00197	.00 00 00 95	.00000 00346	.00 00 00 D5	.00000 00495
.00 00 00 16	.00000 00051	.00 00 00 56	.00000 00200	.00 00 00 96	.00000 00349	.00 00 00 D6	.00000 00498
.00 00 00 17	.00000 00053	.00 00 00 57	.00000 00202	.00 00 00 97	.00000 00351	.00 00 00 D7	.00000 00500
.00 00 00 18	.00000 00055	.00 00 00 58	.00000 00204	.00 00 00 98	.00000 00353	.00 00 00 D8	.00000 00502
.00 00 00 19	.00000 00058	.00 00 00 59	.00000 00207	.00 00 00 99	.00000 00356	.00 00 00 D9	.00000 00505
.00 00 00 1A	.00000 00060	.00 00 00 5A	.00000 00209	.00 00 00 9A	.00000 00358	.00 00 00 DA	.00000 00507
.00 00 00 1B	.00000 00062	.00 00 00 5B	.00000 00211	.00 00 00 9B	.00000 00360	.00 00 00 DB	.00000 00509
.00 00 00 1C	.00000 00065	.00 00 00 5C	.00000 00214	.00 00 00 9C	.00000 00363	.00 00 00 DC	.00000 00512
.00 00 00 1D	.00000 00067	.00 00 00 5D	.00000 00216	.00 00 00 9D	.00000 00365	.00 00 00 DD	.00000 00514
.00 00 00 1E	.00000 00069	.00 00 00 5E	.00000 00218	.00 00 00 9E	.00000 00367	.00 00 00 DE	.00000 00516
.00 00 00 1F	.00000 00072	.00 00 00 5F	.00000 00221	.00 00 00 9F	.00000 00370	.00 00 00 DF	.00000 00519
.00 00 00 20	.00000 00074	.00 00 00 60	.00000 00223	.00 00 00 A0	.00000 00372	.00 00 00 E0	.00000 00521
.00 00 00 21	.00000 00076	.00 00 00 61	.00000 00225	.00 00 00 A1	.00000 00374	.00 00 00 E1	.00000 00523
.00 00 00 22	.00000 00079	.00 00 00 62	.00000 00228	.00 00 00 A2	.00000 00377	.00 00 00 E2	.00000 00526
.00 00 00 23	.00000 00081	.00 00 00 63	.00000 00230	.00 00 00 A3	.00000 00379	.00 00 00 E3	.00000 00528
.00 00 00 24	.00000 00083	.00 00 00 64	.00000 00232	.00 00 00 A4	.00000 00381	.00 00 00 E4	.00000 00530
.00 00 00 25	.00000 00086	.00 00 00 65	.00000 00235	.00 00 00 A5	.00000 00384	.00 00 00 E5	.00000 00533
.00 00 00 26	.00000 00088	.00 00 00 66	.00000 00237	.00 00 00 A6	.00000 00386	.00 00 00 E6	.00000 00535
.00 00 00 27	.00000 00090	.00 00 00 67	.00000 00239	.00 00 00 A7	.00000 00388	.00 00 00 E7	.00000 00537
.00 00 00 28	.00000 00093	.00 00 00 68	.00000 00242	.00 00 00 A8	.00000 00391	.00 00 00 E8	.00000 00540
.00 00 00 29	.00000 00095	.00 00 00 69	.00000 00244	.00 00 00 A9	.00000 00393	.00 00 00 E9	.00000 00542
.00 00 00 2A	.00000 00097	.00 00 00 6A	.00000 00246	.00 00 00 AA	.00000 00395	.00 00 00 EA	.00000 00544
.00 00 00 2B	.00000 00100	.00 00 00 6B	.00000 00249	.00 00 00 AB	.00000 00398	.00 00 00 EB	.00000 00547
.00 00 00 2C	.00000 00102	.00 00 00 6C	.00000 00251	.00 00 00 AC	.00000 00400	.00 00 00 EC	.00000 00549
.00 00 00 2D	.00000 00104	.00 00 00 6D	.00000 00253	.00 00 00 AD	.00000 00402	.00 00 00 ED	.00000 00551
.00 00 00 2E	.00000 00107	.00 00 00 6E	.00000 00256	.00 00 00 AE	.00000 00405	.00 00 00 EE	.00000 00554
.00 00 00 2F	.00000 00109	.00 00 00 6F	.00000 00258	.00 00 00 AF	.00000 00407	.00 00 00 EF	.00000 00556
.00 00 00 30	.00000 00111	.00 00 00 70	.00000 00260	.00 00 00 B0	.00000 00409	.00 00 00 F0	.00000 00558
.00 00 00 31	.00000 00114	.00 00 00 71	.00000 00263	.00 00 00 B1	.00000 00412	.00 00 00 F1	.00000 00561
.00 00 00 32	.00000 00116	.00 00 00 72	.00000 00265	.00 00 00 B2	.00000 00414	.00 00 00 F2	.00000 00563
.00 00 00 33	.00000 00118	.00 00 00 73	.00000 00267	.00 00 00 B3	.00000 00416	.00 00 00 F3	.00000 00565
.00 00 00 34	.00000 00121	.00 00 00 74	.00000 00270	.00 00 00 B4	.00000 00419	.00 00 00 F4	.00000 00568
.00 00 00 35	.00000 00123	.00 00 00 75	.00000 00272	.00 00 00 B5	.00000 00421	.00 00 00 F5	.00000 00570
.00 00 00 36	.00000 00125	.00 00 00 76	.00000 00274	.00 00 00 86	.00000 00423	.00 00 00 F6	.00000 00572
.00 00 00 37	.00000 00128	.00 00 00 77	.00000 00277	.00 00 00 B7	.00000 00426	.00 00 00 F7	.00000 00575
.00 00 00 38	.00000 00130	.00 00 00 78	.00000 00279	.00 00 00 B8	.00000 00428	.00 00 00 F8	.00000 00577
.00 00 00 39	.00000 00132	.00 00 00 79	.00000 00281	.00 00 00 B9	.00000 00430	.00 00 00 F9	.00000 00579
.00 00 00 3A	.00000 00135	.00 00 00 7A	.00000 00284	.00 00 00 BA	.00000 00433	.00 00 00 FA	.00000 00582
.00 00 00 3B	.00000 00137	.00 00 00 7B	.00000 00286	.00 00 00 BB	.00000 00435	.00 00 00 FB	.00000 00584
.00 00 00 3C	.00000 00139	.00 00 00 7C	.00000 00288	.00 00 00 BC	.00000 00437	.00 00 00 FC	.00000 00586
.00 00 00 3D	.00000 00142	.00 00 00 7D	.00000 00291	.00 00 00 BD	.00000 00440	.00 00 00 FD	.00000 00589
.00 00 00 3E	.00000 00144	.00 00 00 7E	.00000 00293	.00 00 00 BE	.00000 00442	.00 00 00 FE	.00000 00591
.00 00 00 3F	.00000 00146	.00 00 00 7F	.00000 00295	.00 00 00 BF	.00000 00444	.00 00 00 FF	.00000 00593

MATHEMATICAL CONSTANTS

Constant	Decimal Value	Hexadecimal Value
π	3.14159 26535 89793	3.243F 6A89
π^{-1}	0.31830 98861 83790	0.517C C1B7
$\sqrt{\pi}$	1.77245 38509 05516	1.C5BF 891C
$\ln \pi$	1.14472 98858 49400	1.250D 048F
e	2.71828 18284 59045	2.B7E1 5163
e^{-1}	0.36787 94411 71442	0.5E2D 58D9
$\sqrt{e}$	1.64872 12707 00128	1.A612 98E2
$\log_{10} e$	0.43429 44819 03252	0.6F2D EC55
$\log_2 e$	1.44269 50408 88963	1.7154 7653
γ	0.57721 56649 01533	0.93C4 67E4
$\ln \gamma$	-0.54953 93129 81645	-0.8CAE 9BC1
$\sqrt{2}$	1.41421 35623 73095	1.6A09 E668
$\ln 2$	0.69314 71805 59945	0.B172 17F8
$\log_{10} 2$	0.30102 99956 63981	0.4D10 4D42
$\sqrt{10}$	3.16227 76601 68379	3.298B 075C
$\ln 10$	2.30258 50929 94046	2.4D76 3777

TABLE OF POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625

ASCII TELETYPE CODES

Symbol	Hexadecimal Code	Symbol	Hexadecimal Code
@	C0	b̸	A0
A	C1	!	A1
B	C2	"	A2
C	C3	#	A3
D	C4	$	A4
E	C5	%	A5
F	C6	&	A6
G	C7	'	A7
H	C8	(	A8
I	C9	)	A9
J	CA	*	AA
K	CB	+	AB
L	CC	,	AC
M	CD	-	AD
N	CE	.	AE
O	CF	/	AF
P	D0	0	B0
Q	D1	1	B1
R	D2	2	B2
S	D3	3	B3
T	D4	4	B4
U	D5	5	B5
V	D6	6	B6
W	D7	7	B7
X	D8	8	B8
Y	D9	9	B9
Z	DA	:	BA
[	DB	;	BB
\	DC	<	BC
]	DD	=	BD
↑	DE	>	BE
←	DF	?	BF

NULL	00
BELL	87
CR	8D
LF	8A
RUBOUT	FF

816 INSTRUCTION SET - NUMERICAL ORDER

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
0000	NOP	No Operation	1
0008	XRM	Set X register to minus 1	1
0010	ARM	Set A register to minus 1	1
0018	AXM	Set A and X register to minus 1	1
0030	TXA	Transfer X to A	1
0048	TAX	Transfer A to X	1
0068	ANX	AND of A and X to X	1
0070	ANA	AND of A and X to A	1
0078	ANB	AND of A and X to A and X	1
0088		Set X to minus 2	1
0090		Set A to minus 2	1
0098		Set A and X to minus 2	1
00A8	DXR	Decrement X register	1
00B0	DXA	Decrement X and put in A	1
00B8	DXB	Decrement X and Put in A and X	1
00C8	DAX	Decrement A and put in X	1
00D0	DAR	Decrement A register	1
00D8	DAB	Decrement A and put in A and X	1
00E8		AND of A and X-1 to X	1
00F0		AND of A and X-1 to A	1
00F8		AND of A and X-1 to A and X	1
0108	ZXR	Zero X register	1
0110	ZAR	Zero A register	1
0018	ZAX	Zero A and X register	1
0128	IXR	Increment X register	1
0130	IXA	Increment X and put in A	1
0138	IXB	Increment X and put in A and X	1
0148	IAX	Increment A and put in X	1
0150	IAR	Increment A register	1
0158	IAB	Increment A and put in A and X	1
0168		AND of A and X +1 to X	1
0170		AND of A and X+1 to A	1
0178		AND of A and X+1 to A and X	1
0208	CAX	Complement A and put in X	1
0210	CAR	Complement A register	1
0218	CAB	Complement A and put in A and X	1
0228		Complement of A ANDed with X to X	1
0230		Complement of A ANDed with X to A	1
0238		Complement of A ANDed with X to A and X	1
0288		A-2 to X	1
0290		A-2 to A	1
0298		A-2 to A and X	1

816 Instructions - Numerical Order (continued)

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
02A8		Complement of A ANDed with X-1 to X	1
02B0		Complement of A ANDed with X-1 to A	1
02B8		Complement of A ANDed with X-1 to A and X	1
0308	NAX	Negate A and put in X	1
0310	NAR	Negate A register	1
0318	NAB	Negate A and put in A and X	1
0328		Complement of A ANDed with X+1 to X	1
0330		Complement of A ANDed with X+1 to A	1
0338		Complement of A ANDed with X+1 to A and X	1
0350	ARP	Set A register to plus 1	1
0358	AXP	Set A and X register to plus 1	1
0408	CXR	Complement X register	1
0410	CXA	Complement X and put in A	1
0418	CXB	Complement X and put in A and X	1
0448		A ANDed with complement of X to X	1
0450		A ANDed with complement of X to A	1
0458		A ANDed with complement of X to A and X	1
0488		X-2 to X	1
0490		X-2 to A	1
0498		X-2 to A and X	1
04C8		A ANDed with complement of (X)-1 to X	1
04D0		A ANDed with complement of (X)-1 to A	1
04D8		A ANDed with complement of (X)-1 to A and X	1
0508	NXR	Negate X register	1
0510	NXA	Negate X and put in A	1
0518	NXB	Negate X and put in A and X	1
0528	XRP	Set X register to plus 1	1
0548		A ANDed with complement of (X)+1 to X	1
0550		A ANDed with complement of (X)+1to A	1
0058		A ANDed with complement of (X)+1 to A and X	1
0608	NRX	NOR of (A and X) to X	1
0610	NRA	NOR of (A and X) to A	1
0618	NRB	NOR of (A and X) to A and X	1
0688		NOR of (A and X)-1 to X	1
0690		NOR of (A and X)-1 to A	1
0698		NOR of (A and X)-1 to A and X	1
0708		NOR of (A and X)+1 to X	1
0710		NOR of (A and X)+1 to A	1
0718		NOR of (A and X)+1 to A and X	1
0080*	HLT	Halt	1
0A00*	EIN	Enable interrupts	1
0C00*	DIN	Disable interrupts	1

*These codes may be added to any OP Code less than 0200 to define a multi-condition operation.

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
1028	ALX	Arithmetic shift X left	1+25K
1050	ALA	Arithmetic shift A left	1+25K
10A8	ARX	Arithmetic shift X right	1+25K
10D0	ARA	Arithmetic shift A right	1+25K
1128	RLX	Rotate X left with OV	1+25K
1150	RLA	Rotate A left with OV	1+25K
11A8	RRX	Rotate X right with OV	1+25K
11D0	RRA	Rotate A right with OV	1+25K
1200	ROV	Reset overflow	1
1328	LLX	Logical shift X left	1+25K
1350	LLA	Logical shift A left	1+25K
13A8	LRX	Logical shift X right	1+25K
13D0	LRA	Logical shift A right	1+25K
1400	SOV	Set overflow	1
1600	COV	Complement overflow	1
1980	LRR	Long rotate right A and X	1+25K
1900	LRL	Long rotate left A and X	1+25K

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
2080	JAM	Jump forward if A negative	1
2100	JAZ	Jump forward if A zero	1
2180	JAL	Jump forward if A negative or equal zero	1
2200	JOS	Jump forward if overflow set	1
2280		Jump forward if X equal 1 or A negative	1
2300		Jump forward if X equal 1 or A equal zero	1
2380		Jump forward if X equal 1 or A negative or equal to zero	1
2400	JSR	Jump forward if SS off	1
2480		Jump forward if SS off or A negative	1
2500		Jump forward if SS off or A equal to zero	1
2580		Jump forward if SS off or A negative or equal to zero	1
2600		Jump forward if SS off or OV set	1
2680		Jump forward if SS off or OV set or A negative	1
2700		Jump forward if SS off or OV set or A equal zero	1
2780		Jump forward if SS off or OV reset or A less than or equal to zero	1
2800	JXZ	Jump forward if X equal zero	1
2880		Jump forward if X equal zero or A negative	1
2900		Jump forward if X equal zero or A equal zero	1
2980		Jump forward if X equal zero or A negative equal zero	1
2A00		Jump forward if X equal zero or OV set	1
2A80		Jump forward if X equal zero or OV set or A negative	1
2B00		Jump forward if X equal zero or OV set or A equal zero	1
2B80		Jump forward if X equal zero or OV set or Anegative equal to zero	1
2C00		Jump forward if X equal zero or SS off	1
2C80		Jump forward if X equal zero or SS off or A negative	1
2D00		Jump forward if X equal zero or SS off or A equal zero	1
2D80		Jump forward if X equal zero or SS off or A negative or equal zero	1
2E00		Jump forward if X equal zero or SS off or OV set	1
2E80		Jump forward if X equal or SS off or OV set or A negative	1
2F00		Jump forward if X equal zero or SS off or OV set or A equal to zero	1
2F80		Jump forward if X equal zero or SS off or OV set or A negative	1

816 Instructions - Numerical Order (continued)

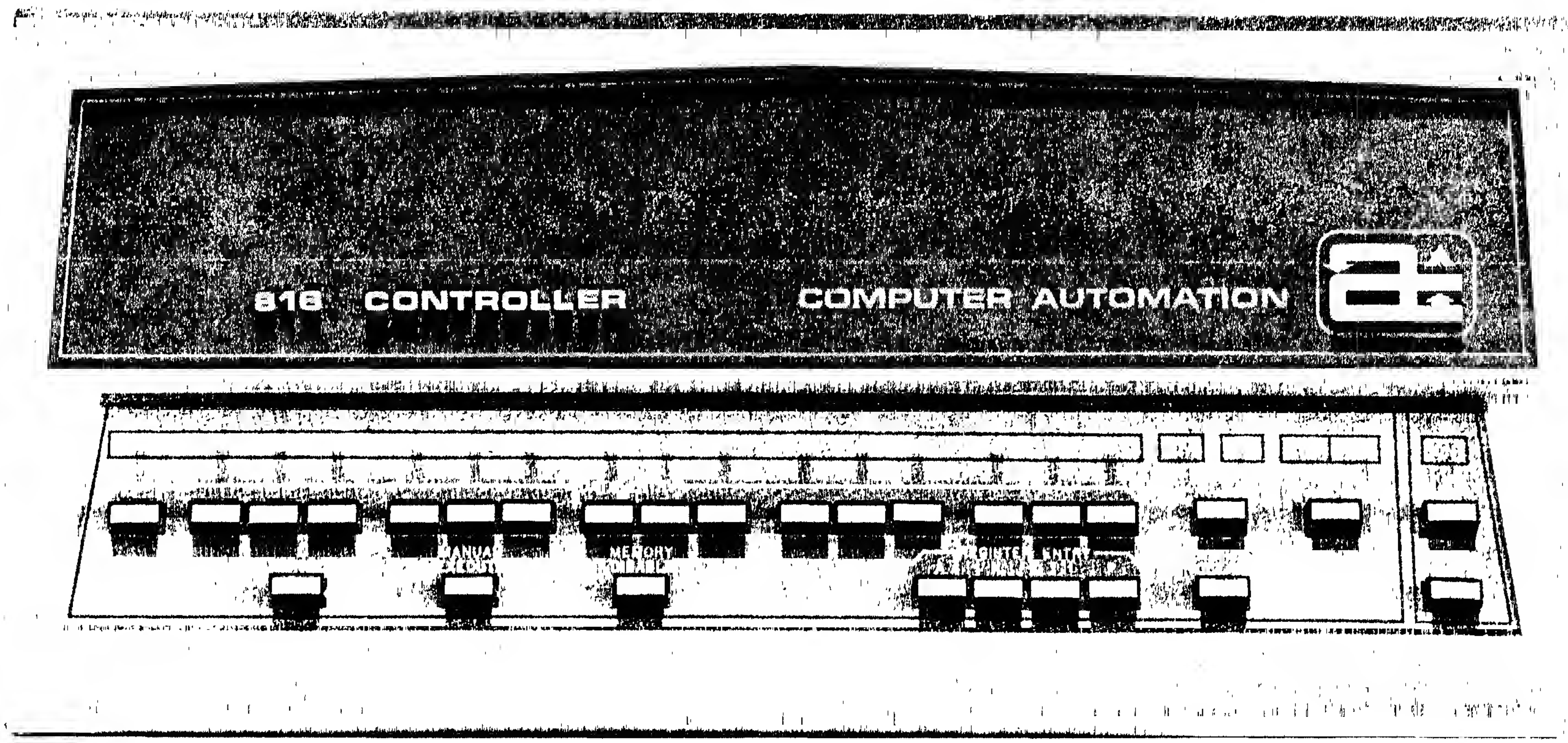
Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
3080	JAP	Jump forward if A positive or equal to zero	1
3100	JAN	Jump forward if A not zero	1
3180	JAG	Jump forward if A positive and not equal to zero	1
3200	JOR	Jump forward if OV reset	1
3280		Jump forward if A positive and OV reset	1
3300		Jump forward if A non-zero and OV reset	1
3380		Jump forward if A non-zero and positive and OV reset	1
3400	JSS	Jump forward if SS on	1
3480		Jump forward if SS on and A positive	1
3500		Jump forward if SS on and A non-zero	1
3580		Jump forward if SS on and A positive and non-zero	1
3600		Jump forward if SS on and OV reset	1
3680		Jump forward if SS on and A positive and OV reset	1
3700		Jump forward if SS on and A non-zero and OV reset	1
3780		Jump forward if A non-zero and positive and SS on and OV reset	1
3800	JXN	Jump forward if X non-zero	1
3880		Jump forward if X non-zero and A positive	1
3900		Jump forward if X non-zero and A non-zero	1
3980		Jump forward if X non-zero and A positive and non-zero	1
3A00		Jump forward if X non-zero and OV reset	1
3A80		Jump forward if X non-zero and A positive and OV reset	1
3B00		Jump forward if X non-zero and A non-zero and OV reset	1
3B80		Jump forward if X non-zero and A non-zero and positive and OV reset	1
3C00		Jump forward if X non-zero and SS equal 1	1
3C80		Jump forward if X non-zero and A positive and SS on	1
3D00		Jump forward if X non-zero and A non-zero and SS on	1
3D80		Jump forward if X non-zero and A non-zero and positive and SS on	1
3E00		Jump forward if X non-zero and SS on and OV reset	1
3E80		Jump forward if X non-zero and A positive and SS on and OV reset	1
3F00		Jump forward if X non-zero and A non-zero and SS on and OV reset	1
3F80		Jump forward if X non-zero and A non-zero and positive and SS on and OV reset	1

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
4000	SEL	Select function	1
4800	SSN	Sense and skip on no response	1
4900	SEN	Sense and skip on response	1
5000	INB	Input block to memory	
5800	INA	Input to A register (unconditionally)	1
5900	RDA	Read word to A register	1
5A00	INX	Input to X register	1
5B00	RDX	Read word to X register	1
5C00	INAM	Masked input to A register (unconditionally)	1
5D00	RDAM	Read word to A register masked	1
5E00	INXM	Masked input to X register (unconditionally)	1
5F00	RDXM	Read word to X register masked	1
6000	OTB	Output block from memory	
6800	OTZ	Output zero (unconditionally)	1
6900	WRZ	Write zeros	1
6C00	OTA	Output A register (unconditionally)	1
6D00	WRA	Write from A register	1
6E00	OTX	Output X register (unconditionally)	1
6F00	WRX	Write from X register	1
7100	LDM	Load memory	1
7500	DPM	Dump memory	1
7800	IBA	Input byte to A register (unconditionally)	1
7900	RBA	Read byte to A register	1
7A00	IBX	Input byte to X register (unconditionally)	1
7B00	RBX	Read byte to X register	1
7C00	IBAM	Input byte to A register masked (unconditionally)	1
7D00	RBAM	Read byte to A register masked	1
7E00	IBXM	Input byte to X register masked (unconditionally)	1
7F00	RBXM	Read byte to X register masked	1

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
8000	AND	AND to A, direct	2
8100		AND to A, indirect	3
8200		AND to A relative to P forward	2
8300		AND to A relative to P forward, indirect	3
8400		AND to A indexed	2
8500		AND to A indexed, indirect	3
8600		AND to A relative to P backward	2
8700		AND to A relative to P backward, indirect	3
8800	ADD	Add to A direct	2
8900		Add to A indirect	3
8A00		Add to A relative to P forward	2
8B00		Add to A relative to P forward, indirect	3
8C00		Add to A indexed	2
8D00		Add to A indexed, indirect	3
8E00		Add to A relative to P backward	2
8F00		Add to A relative to P backward, indirect	3
9000	SUB	Subtract from A, direct	2
9100		Subtract from A, indirect	3
9200		Subtract relative to P forward	2
9300		Subtract relative to P forward, indirect	3
9400		Subtract from A, indexed	2
9500		Subtract from A, indexed, indirect	3
9600		Subtract, relative to P backward	2
9700		Subtract, relative to P backward, indirect	3
9800	STA	Store A direct	2
9900		Store A indirect	3
9A00		Store A relative to P forward	2
9B00		Store A relative to P forward, indirect	3
9C00		Store A, indexed	2
9D00		Store A indexed, indirect	3
9E00		Store A relative to P backward	2
9F00		Store A relative to P backward, indirect	3
A000	IOR	Inclusive OR to A, direct	2
A100		Inclusive OR to A, indirect	3
A200		Inclusive OR to A, relative to P forward	2
A300		Inclusive OR to A, relative to P forward, indirect	3
A400		Inclusive OR to A, indexed	2
A500		Inclusive OR to A, indexed, indirect	3
A600		Inclusive OR to A, relative to P backward	2
A700		Inclusive OR to A, relative to P backward, indirect	3
A800	XOR	Exclusive OR to A, direct	2
A900		Exclusive OR to A, indirect	3
AA00		Exclusive OR to A, relative to P forward	2
AB00		Exclusive OR to A, relative to P forward, indirect	3

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
AC00	XOR	Exclusive OR to A, Indexed	2
AD00		Exclusive OR to A, indexed, indirect	3
AE00		Exclusive OR to A, relative to P backward	2
AF00		Exclusive OR to A, relative to P backward, indirect	3
B000	LDA	Load A, direct	2
B100		Load A, indirect	3
B200		Load A, relative to P foreard	2
B300		Load A, relative to P forward, indirect	3
B400		Load A, indexed	2
B500		Load A, indexed, indirect	3
B600		Load A, relative to P backward	2
B700		Load A, relative to P backward, indirect	3
B800	EMA	Exchange memory and A, direct	2
B900		Exchange memory and A, indirect	3
BA00		Exchange memory and A, relative to P forward	2
BB00		Exchange memory and A, relative to P forward, indirect	3
BC00		Exchange memory and A, indexed	2
BD00		Exchange memory and A, indexed, indirect	3
BE00		Exchange memory and A, relative to P backward	2
BF00		Exchange memory and A, relative to P backward, indirect	3
C000	CAI	Compare to A immediate	2
C100	CXI	Compare to X immediate	3
C200	ADI	Add to X immediate	2
C300	SBI	Subtract from X immediate	3
C400	LXP	Load X positive immediate	2
C500	LXM	Load X minus immediate	3
C600	LAP	Load A positive immediate	2
C700	LAM	Load A minus immediate	3
C800	SCN	Scan memory, direct	2
C900		Scan memory, indirect	3
CA00		Scan memory, relative to P forward	2
CB00		Scan memory, relative to P forward, indirect	3
CC00		Scan memory, indexed	2
CD00		Scan memory, indexed, indirect	3
CE00		Scan memory, relative to P backward	2
CF00		Scan memory, relative to P backward, indirect	3
D000	CMS	Compare and skip if high or equal, direct	2
D100		Compare and skip if high or equal, indirect	3
D200		Compare and skip if high or equal, relative to P forward	2
D300		Compare and skip if high or equal, relative to P forward, indirect	3
D400		Compare and skip if high or equal, indexed	2
D500		Compare and skip if high or equal, indexed, indirect	3
D600		Compare and skip if high or equal, relative to P backward	2
D700		Compare and skip if high or equal, relative to P backward, indirect	3

Instruction Code In Hexadecimal	Instruction Mnemonic	Name	Cycles
D800	IMS	Increment memory and skip on zero result, direct	2
D900		Increment memory and skip on zero result, indirect	3
DA00		Increment memory and skip on zero, relative to P forward	2
DB00		Increment memory and skip on zero, relative to P forward, indirect	3
DC00		Increment memory and skip on zero, indexed	2
DD00		Increment memory and skip on zero, indexed, indirect	3
DE00		Increment memory and skip on zero, relative to P backward	2
DF00		Increment memory and skip on zero relative to P backward, indirect	3
E000	LDX	Load X, direct	2
E100		Load X, indirect	3
E200		Load X, relative to P forward	2
E300		Load X, relative to P forward, indirect	3
E400		Load X, Indexed	2
E500		Load X, indexed, indirect	3
E600		Load X, relative to P backward	2
E700		Load X, relative to P backward, indirect	3
E800	STX	Store X, direct	2
E900		Store X, indirect	3
EA00		Store X, relative to P forward	2
EB00		Store X, relative to P forward, indirect	3
EC00		Store X, indexed	2
ED00		Store X, indexed, indirect	3
EE00		Store X, relative to P backward	2
EF00		Store X, relative to P backward, indirect	3
F000	JMP	Jump unconditionally, direct	2
F100		Jump unconditionally, indirect	3
F200		Jump unconditionally, relative to P forward	2
F300		Jump unconditionally, relative to P forward, indirect	3
F400		Jump unconditionally, indexed	2
F500		Jump unconditionally, indexed, indirect	3
F600		Jump unconditionally, relative to P backward	2
F700		Jump unconditionally, relative to P backward, indirect	3
F800	JST	Jump and store, direct	2
F900		Jump and store, indirect	3
FA00		Jump and store, relative to P forward	2
FB00		Jump and store, relative to P forward, indirect	3
FC00		Jump and store, indexed	2
FD00		Jump and store, indexed, indirect	3
FE00		Jump and store, relative to P backward	2
FF00		Jump and store, relative to P backward, indirect	3



Control Console

CONSOLE DISPLAY PROCEDURE

1. Place the STEP/RUN switch in the STEP mode.

2. Check the MEMORY DISABLE switch to be sure it is reset (not depressed).

3. Depress the Instruction REGISTER ENTRY switch (I), depress the CLEAR switch, and enter B600 on the DATA ENTRY switches. The B600 instruction loads the Accumulator register (A), relative to the Program Counter register (P).

4. Set (depress) the MANUAL EXECUTE switch. This prevents the processor from executing the instruction fetch cycle, effectively "locking" the load instruction into the Instruction register.

5. Depress the Program Counter REGISTER ENTRY switch (P), depress the CLEAR switch, and enter on the DATA ENTRY switches the address of the memory word to be displayed.

6. Depress the Accumulator REGISTER ENTRY switch (A).

7. Depress the CYCLE switch. The contents of the memory word specified by the Program Counter register (P) will be loaded into the Accumulator register (A) and displayed on the REGISTER DISPLAY lights, and the Program Counter will be incremented by one (P = P+1).

8. Repeat step 7 for each successive memory word to be displayed.

9. To display data from a new location, go to step 5.

CONSOLE LOAD PROCEDURE

1. Place the STEP/RUN switch in the STEP mode.

2. Check the MEMORY DISABLE switch to be sure it is reset (not depressed).

3. Depress the Instruction REGISTER ENTRY switch (I), depress the CLEAR switch, and enter 9E00 on the DATA ENTRY switches. The 9E00 instruction stores the Accumulator register (A), relative to the Program Counter register (P).

4. Set (depress) the MANUAL EXECUTE switch. This prevents the processor from executing the instruction fetch cycle, effectively "locking" the store instruction into the Instruction register.

5. Depress the Program Counter REGISTER ENTRY switch (P), depress the CLEAR switch, and enter on the DATA ENTRY switches the address of the word to be loaded into memory.

6. Depress the Accumulator REGISTER ENTRY switch (A).

7. Depress the CLEAR switch and enter on the DATA ENTRY switches the instruction or data to be loaded into memory.

8. Depress the CYCLE switch. The contents of the Accumulator register (A) will be loaded into memory at the location specified by the contents of the Program Counter register (P), and the Program Counter register will be incremented by one (P = P+1).

9. Repeat steps 7 and 8 for each successive instruction or data word to be loaded into memory.

10. To load data at a new location, go to step 5.

COMPUTER AUTOMATION INCORPORATED
895 WEST SIXTEENTH STREET
NEWPORT BEACH, CALIF. 92660
TELEPHONE· (714) 642-9630